NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

VERTICAL PLANE OBSTACLE AVOIDANCE AND CONTROL OF THE REMUS AUTONOMOUS UNDERWATER VEHICLE USING FORWARD LOOK SONAR

by

Daniel L. Hemminger

June 2005

Thesis Advisor:	Anthony Healey
Second Reader:	Doug Horner

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2005	**3. REPORT TYPE AND DATES COVERED** Master's Thesis

4. TITLE AND SUBTITLE: Title (Mix case letters) Vertical Plane Obstacle Avoidance and Control of the REMUS Autonomous Underwater Vehicle Using Forward Look Sonar	**5. FUNDING NUMBERS**
6. AUTHOR(S) Daniel Hemminger	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

Current rates of technological advancement continue to translate into changes on our battlefields. Aerial robots capable of gathering reconnaissance along with unmanned underwater vehicles capable of defusing enemy minefields provide evidence that machines are playing key roles once played by humans within our military. This thesis explores one of the major problems facing both commercial and military UUVs to date. Successfully navigating in unfamiliar environments and maneuvering autonomously to avoid obstacles is a problem that has yet to be fully solved. Using a simulated 2-D ocean environment, the work of this thesis provides results of numerous REMUS simulations that model the vehicle's flight path over selected sea bottoms. Relying on a combination of sliding mode control and feedforward preview control, REMUS is able to locate obstacles such as seawalls using processed forward look sonar images. Once recognized, REMUS maneuvers to avoid the obstacle according to a Gaussian potential function. In summary, the integration of feedforward preview control and sliding mode control results in an obstacle avoidance controller that is not only robust, but also autonomous.

14. SUBJECT TERMS REMUS, AUV, UUV, autonomy, obstacle avoidance, vertical plane, sliding mode controller, feedforward preview controller, forward look sonar	**15. NUMBER OF PAGES** 97
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

THIS PAGE INTENTIONALLY LEFT BLANK

VERTICAL PLANE OBSTACLE AVOIDANCE AND CONTROL OF THE REMUS AUTONOMOUS UNDERWATER VEHICLE USING FORWARD LOOK SONAR

Daniel L. Hemminger
Ensign, United States Navy
B.S., United States Naval Academy, 2004

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
JUNE 2005

Author: Daniel L. Hemminger

Approved by: Anthony J. Healey
Thesis Advisor

Doug Horner
Second Reader

Anthony J. Healey
Chairman, Department of Mechanical Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Current rates of technological advancement continue to translate into changes on our battlefields. Aerial robots capable of gathering reconnaissance along with unmanned underwater vehicles capable of defusing enemy minefields provide evidence that machines are playing key roles once played by humans within our military. This thesis explores one of the major problems facing both commercial and military UUVs to date. Successfully navigating in unfamiliar environments and maneuvering autonomously to avoid obstacles is a problem that has yet to be fully solved. Using a simulated 2-D ocean environment, the work of this thesis provides results of numerous REMUS simulations that model the vehicle's flight path over selected sea bottoms. Relying on a combination of sliding mode control and feedforward preview control, REMUS is able to locate obstacles such as seawalls using processed forward look sonar images. Once recognized, REMUS maneuvers to avoid the obstacle according to a Gaussian potential function. In summary, the integration of feedforward preview control and sliding mode control results in an obstacle avoidance controller that is not only robust, but also autonomous.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION....................1
  A. PERSPECTIVE....................1
  B. BACKGROUND....................1
  C. PLATFORM....................3
  D. APPROACH....................5

II. KINEMATICS AND DYNAMICS....................7
  A. ASSUMPTIONS....................7
  B. COORDINATE SYSTEM TRANSFORMATION....................7
  C. LINEAR AND ROTATIONAL VELOCITY....................9
  D. EQUATIONS OF MOTION....................10
  E. VERTICAL PLANE SIMPLIFICATIONS....................12

III. CONTROL METHODS....................15
  A. SLIDING MODE CONTROLLER / FEEDFORWARD PREVIEW CONTROLLER....................15

IV. SPACE SIMULATION....................21
  A. 2-D OCEAN ENVIRONMENT MODEL....................21
    1. Sea Floor Model: Gradual Rise and Step....................21
    2. Sea Floor Model: Hill and Wall....................23
  B. STATE INTEGRATION....................24
  C. SONAR MODEL....................25

V. VEHICLE SIMULATION....................29
  A. INITIAL TESTS....................29
  B. OBSTACLE AVOIDANCE TESTING....................31
    1. Optimizing Obstacle Avoidance through Gaussian Potential Field Sizing....................32
    2. REMUS’ Dynamic Response Over a 6 Meter Gradual Incline....................35
    3. REMUS’ Dynamic Response Over a 6 Meter Sharp Step....................39
    4. REMUS’ Dynamic Response Over a 6 Meter Sea Floor Hill....................43
    5. REMUS’ Dynamic Response Over 6 Meter Sea Wall....................46

VI. CONCLUSION AND RECOMMENDATIONS....................53
  A. CONCLUSION....................53
  B. RECOMMENDATIONS....................53

APPENDIX I: MATLAB CODES FOR GRADUAL RISE/STEP USING FULL POTENTIAL FUNCTION....................55

APPENDIX II: MATLAB CODES FOR GRADUAL RISE/STEP USING HALF POTENTIAL FUNCTION....................63

APPENDIX III: MATLAB CODES FOR HILL/SEA WALL USING FULL POTENTIAL FUNCTION.....71

LIST OF REFERENCES.....79

INITIAL DISTRIBUTION LIST.....81

LIST OF FIGURES

Figure 1. REMUS Vehicle 3
Figure 2. Rotational Transformation using Euler Angles 8
Figure 3. Block Diagram of Combination of Feedforward Preview Controller and Sliding Mode Altitude Controller 17
Figure 4. Gaussian Potential Field Boundary Around Sea Bottom Rise.........................18
Figure 5. Simulated Gradual Rise Sea Bottom...22
Figure 6. Simulated Sharp Step Sea Bottom 22
Figure 7. Simulated Hill Sea Bottom 23
Figure 8. Simulated Seawall...24
Figure 9. REMUS Sonar Schematic: RDI Doppler and Forward Look Sonar................25
Figure 10. Mechanics Behind REMUS’ Simulated Forward Look Sonar 26
Figure 11. Forward Look Sonar Image of Gradual 6 meter Rise Located at 60 meters Along the Global X axis 27
Figure 12. Graph Comparing Minimum Clearance of REMUS With and Without Forward Look Sonar 29
Figure 13. REMUS Collision: Forward Look Sonar Off 30
Figure 14. REMUS Obstacle Avoidance: Forward Look Sonar On 31
Figure 15. Obstacle Avoidance Depth Commands using Half of a Gaussian Potential Field 32
Figure 16. REMUS Dynamic Response over Gradual Rise Using Half of a Potential Field 33
Figure 17. REMUS Dynamic Response over Sharp Step Using Half of a Potential Field 34
Figure 18. More Gradual Response Achieved through Gaussian Potential Function Variable Manipulation 35
Figure 19. REMUS Dynamic Response Over Gradually Rising Sea Bottom..................36
Figure 20. Sequential Display of Forward Look Sonar Images of Gradual 6 meter Sea Floor Rise...38
Figure 21. REMUS Dynamic Response over Sharp Step 39
Figure 22. Sequential Display of Forward Look Sonar Images over a 6 meter High Sharp Step 41
Figure 23. Dynamic Response of REMUS over 6 meter Sharp Step; No Forward Look Sonar...42
Figure 24. Dynamic Response of REMUS over 6 meter Sea Floor Hill..........................43
Figure 25. Sequential Display of Forward Look Sonar Images over a 6 meter Sea Floor Hill..45
Figure 26. Dynamic Response of REMUS over 6 meter Sea Wall..................................47
Figure 27. Dynamic Response of REMUS over 6 meter Sea Wall with Forward Look Sonar Turned off 48
Figure 28. Sequential Display of Forward Look Sonar Images over 6 meter Sea Wall ...50

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

Table 1. REMUS Specifications 4

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank Professor Anthony Healey for his guidance. His knowledge and experience introduced me to a very interesting field of research. I would also like to thank my dog, Taylor for keeping me company during my writing process.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. PERSPECTIVE

Today, we know more about our moon that orbits a quarter of a million miles from Earth than we do about our planet's oceans. (Samuel W. Bodman, U.S. Department of Commerce Deputy Secretary, 2001) Moreover, roughly 75 % of this Earth's surface is water yet only 5 % of our planet's oceans have been explored. Why, in the era of cutting age space technologies and NASA rover expeditions to Mars, have we failed to fully exploit our ocean's obvious untapped resources? Undiscovered new medicines, precious minerals, oil, alternative energy sources, and new aquatic life are only a few examples of what our oceans have to offer. With so much industrial, medical, and biological potential, our oceans also serve as an extremely important arena for our nation's military. Submarine warfare, mine countermeasures, and littoral dominance all depend on our acute understanding of the surrounding marine environment.

With current levels of technological growth, unmanned undersea vehicles (UUV's) are evolving into platforms capable of performing tasks ranging from exploring the deepest uncharted depths of our oceans to detecting and neutralizing enemy mine fields. According to Barbara Fletcher, project manager at the Space and Naval Warfare Systems Center in San Diego California, a UUV is defined as a "self-propelled submersible whose operation is either fully autonomous (pre-programmed or real time adaptive mission control) or under minimal supervisory control, and is untethered except for data links such as fiber optic cable." (Barbara Fletcher, 2001). A UUV's appeal stems not only from its potential to efficiently gather, store, and transmit data in an underwater environment, but also from its ability to do so without risking human life.

B. BACKGROUND

Avoiding obstacles traditionally has posed problems for both land and marine robotic vehicles. Although land robots are capable of avoiding obstacles using the "stop-back-turn" principle, this maneuver becomes more difficult underwater where much more power is required to first stop a vehicle's forward motion and then keep that vehicle hovered onsite (Healey and Kim, 2000). As a result, marine robots need to adopt different methods when avoiding collisions with obstacles underwater. While the technology exists, we have barely scratched the surface of UUV development. With the turn of the 21st century came the first UUV models suited for commercial use (Blidberg, 2001). Despite this accomplishment, serious problems regarding UUV development remain that need to be addressed if progress is to continue in this field. Problems concerning autonomy and navigation are perhaps most serious and have yet to be fully solved (Blidberg, 2001). Up to this point, when preparing a UUV for a mission, waypoints are programmed into the UUV's onboard computer. These waypoints provide the vehicle with a preset trajectory that ultimately guides the vehicle to a desired end position. Although reliable given a familiar marine environment with obstacle locations known in advance, this approach leads to problems when a UUV is tasked with operating in an environment with which it is unfamiliar. Ideally, a UUV should be capable of adapting and reacting to a changing marine environment while still accomplishing its assigned mission. Such capability demands a certain level of artificial intelligence on the vehicle's part-intelligence characterized by independently recognizing and successfully navigating around obstacles that appear in the vehicle's area of operation.

In terms of where our military currently stands concerning UUV technology, RDML Willaim E. Landay III, Executive Program Officer Littoral and Mine Warfare, in a brief given at the Naval Postgraduate School in spring 2003 stated that UUV models posses limited mission capability and lack the modularity necessary to adapt to different littoral environments (RDML Landay, 2003). In other words, current UUV models deployed by our military are suited for specific tasks and environments only and cannot successfully adjust to unfamiliar marine environments. The goal of our Navy is to develop an "affordable system based on standardized platform designs with modular interchangeable payloads, common control, and netcentric information exchange" (RDML Landay, 2003).

C. PLATFORM

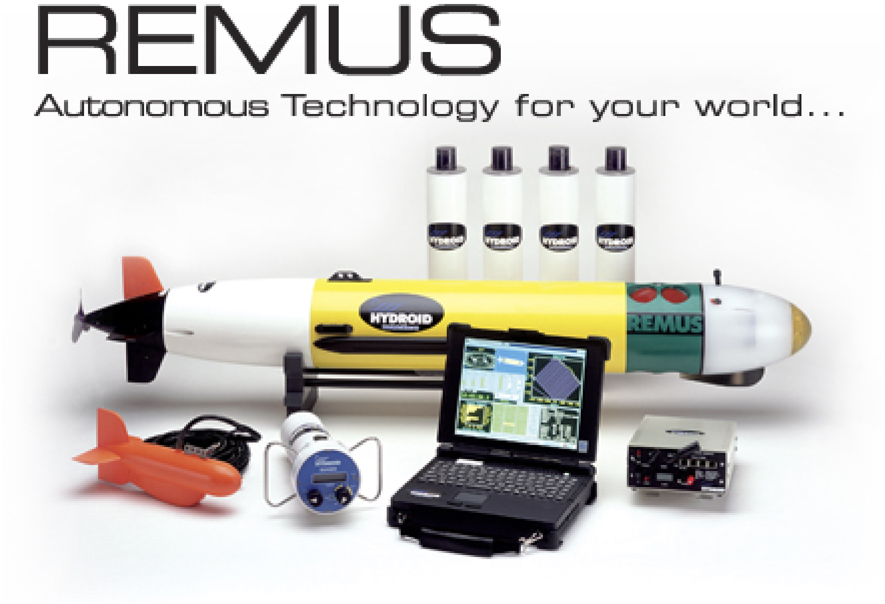


Figure 1. REMUS Vehicle

Currently, the U.S. Navy deploys REMUS (Remote Environmental Monitoring Units) when conducting mine countermeasures (Jordan, 2003). REMUS is an AUV system that was originally developed by Woods Hole Oceanographic Institution and is now commercially manufactured by Hydroid Inc. (Jordan, 2003). With over 60 REMUS systems delivered to date, popularity regarding this torpedo-shaped AUV is quickly growing in both the military and commercial sector (www.hyroidinc.com, 2005). With mission capabilities including environmental monitoring, mine countermeasures, and hydrographic surveying, REMUS, which is less than 4 feet in length and roughly 80 pounds, is an obvious improvement over the previous generation of more cumbersome and less maneuverable AUV systems.

Research concerning AUV technological advancement is being done at The Center for Autonomous Underwater Research in Monterey, California. Professor Anthony J. Healey along with others work with two UUV systems, the Acoustic Radio Interactive Exploratory Server (ARIES), and REMUS. The work of this thesis results from simulations using REMUS only, therefore, ARIES will not be taken into account.

Vehicle Diameter	19 cm
Vehicle Length	160 cm
Weight in Air	37 kg (<80 lbs.)
Trim Weight in Air	1 kg
Maximum Operating Depth	100 meters
Energy	1 kw-hr internally rechargeable Lithium ion
Endurance	22 hours at optimum speed of 1.5 m/s (3 knots). 8 hours at 2.5 m/s (5 knots)
Propulsion	Direct dive DC brushless motor to open three blader propeller
Velocity Range	0.25 to 2.8 m/s variable over range
Control	2 coupled yaw and pitch fins
On/Off	Magnetic switch
External Hook-up	Two pin combined Ethernet, vehicle power and battery charging; 4 pin serial connector
Navigation	Long base line; Ultra short base line; Doppler assisted dead reckon; (Optional: GPS)
Transponders	20-30 kHz operating frequency range
Tracking	Emergency transponder, mission abort, and ORE Trackpoint compatible
Sensors Doppler Velocity Log	RDI 1.2 MHz up/down looking
Side Scan Sonar	600 or 900 kHz MSTL AUV model
Light Scattering Sensor	
Conductivity & Temperature	
Software	GUI based laptop interface for programming, training, post mission analysis, documentation, maintenance and trouble shooting Data exporting and reporting HTML report generator, direct Matlab and ASCII text export Shipping 2 cases for all equipment, each less than 150 lbs (suitable for Fed-Ex transport)

Table 1. REMUS Specifications

REMUS is capable of housing 2 single or dual frequency side scan sonar. This allows the vehicle to sense objects underneath and adjacent to it’s path. Obviously, not having forward look sonar significantly decreases REMUS’ overall sensing capability.

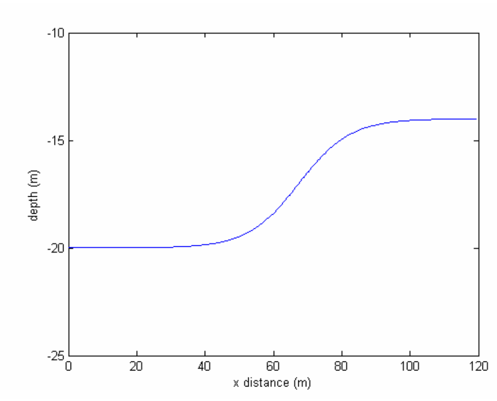
Imagine being asked to drive your car using only your passenger and drivers side windows and not your front windshield. In essence, operating REMUS without forward look sonar is quite similar. Although no forward look sonar is currently attached to the vehicle, Professor Healey along with his research team plan to equip REMUS with forward look capabilities in the near future. This thesis explores the effects forward look sensing capabilities have upon REMUS' ability to recognize and maneuver over obstacles in its path.

D. APPROACH

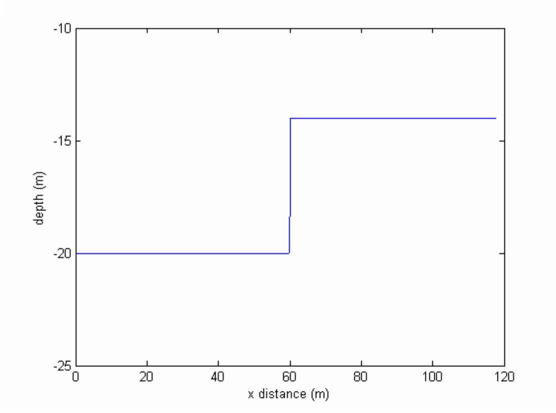
Theses by Fodrea, 2002, and Chuhran, 2003 developed horizontal and vertical plane obstacle avoidance dynamic controllers for REMUS. However, both simulations relied on the fact that the locations of obstacles were known in advance. Contrastingly, the scope of this thesis explores REMUS' ability to autonomously maneuver over obstacles in the vertical plane whose locations were not previously known. In other words, using real time processed forward look sonar images, REMUS will sense an obstacle and then maneuver over that obstacle using its' stern mounted control surfaces.

All simulations for this thesis were done in MATLAB. Real sonar data was not used; rather a sonar model was built in MATLAB, which was capable of gathering 2-D sonar images in the vertical plane. Since REMUS could realistically encounter many different types of sea bottoms during any given mission, this thesis explored four different possible sea bottoms.

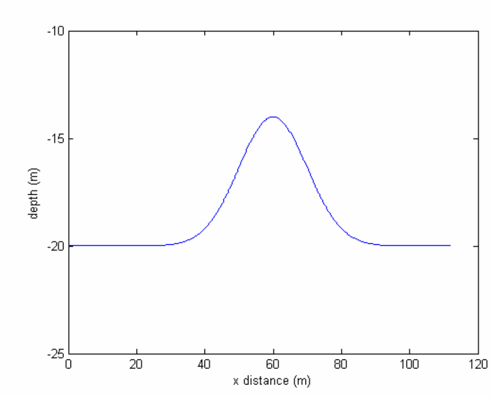
1. Gradual Rise



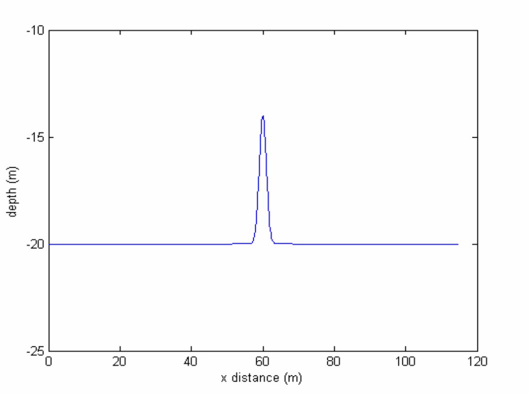
2. Step



3. Hill



4. Wall



Relying on a simulated marine state space model, hydrodynamic equations of motion in the vertical plane coupled with a feedforward preview controller and sliding mode autopilot guided REMUS over the four different sea bottoms.

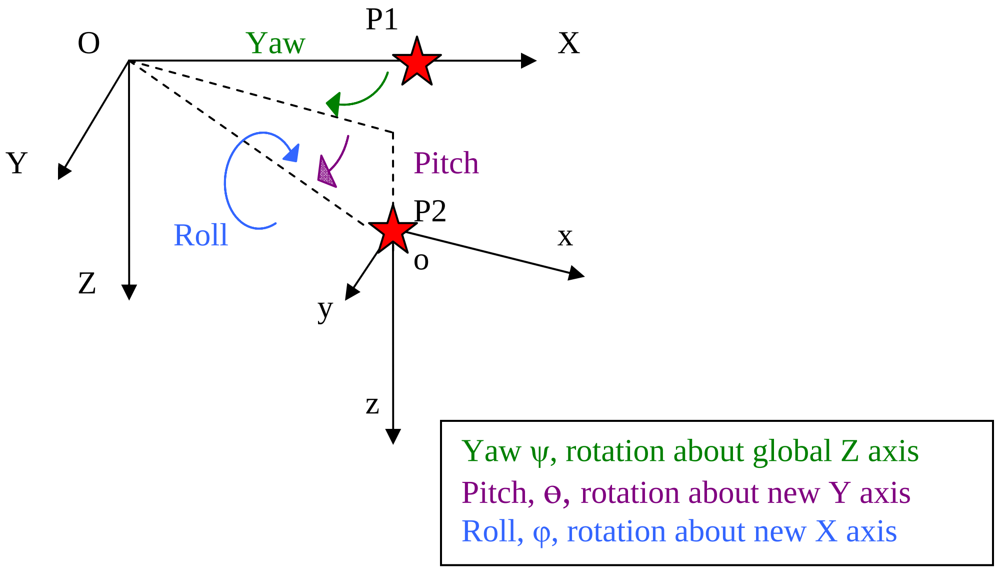
II. KINEMATICS AND DYNAMICS

A. ASSUMPTIONS

For the purpose of this thesis, REMUS will assume rigid body characteristics and will be unable to either flex or deform. This assumption, although not entirely true because all bodies deform when they move, is acceptable noting that any deflections experienced when maneuvering are negligible relative to the overall motion of REMUS. Also, the earth's acceleration is assumed to have a negligible affect on the acceleration components of the vehicle's center of mass (Healey, 2001). Finally, the primary forces that act on the vehicle have inertial, gravitational, hydrostatic, propulsion, thruster, and hydrodynamic forces from lift and drag (Healey, 2001).

B. COORDINATE SYSTEM TRANSFORMATION

The global reference frame represents the entire ocean environment and is defined using the following coordinate system; OXYZ with O placed at the origin and directions north (X), east (Y), and down (Z). Since REMUS operates in a body-fixed or local coordinate system, we define the vehicle's orientation with respect to the global reference frame using 3 rotation angles. These angles are commonly known as Euler angles and represent rotations from the global reference frame OXYZ to a local coordinate system defined oxyz.



O
Yaw
P1
X
Y
Pitch
P2
x
Roll
o
Z
y
z
Yaw ψ, rotation about global Z axis
Pitch, ɵ, rotation about new Y axis
Roll, φ, rotation about new X axis

Figure 2. Rotational Transformation using Euler Angles

Figure 2 displays two points in different coordinate systems. Let us assume that P1 is located in a global reference frame while P2 is located in a hypothetical local reference frame. With a rotational transformation using Euler angles φ, ɵ, ψ, we can easily describe the angular orientation of P2 with respect to P1 and the global reference frame at any point in time. First, we perform an azimuth rotation (φ) around the global Z axis. Next, we perform an elevation rotation around a new Y axis. And lastly, we rotate around a new X axis which defines the vehicle's spin or tilt with respect to the global reference frame. Regarding submersibles, these Euler angle rotations are commonly referred to as yaw, pitch, and roll respectively. The following transformation matrix taken from Professor Healey's notes allows for vehicle position translation from both the global to local reference frame and the local to global reference frame.

$$T(\varphi,\theta,\psi)=\begin{bmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi-\sin\psi\cos\phi & \cos\psi\sin\theta\cos\phi+\sin\psi\sin\phi \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\phi+\cos\psi\cos\phi & \sin\psi\sin\theta\cos\phi-\cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix}$$

C. LINEAR AND ROTATIONAL VELOCITY

In terms of REMUS' motion, translational and rotational velocities need to be defined. Regarding translational motion, REMUS operates under 3 degrees of freedom in 3-D space and thus is capable of 3 different linear velocities;

- Surge velocity (u) along the positive x axis
- Side slip velocity (v) along the positive y axis
- Heave velocity (w) along the positive z axis

Defining the global velocity vector as $[\dot{X} ; \dot{Y} ; \dot{Z}]$ and the local or body-fixed velocity vector as [u ; v ; w], the following transformation links global linear velocities with local linear velocities.

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = T(\phi, \theta, \psi) \bullet \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix}$$

Regarding rotational motion, REMUS operates under 3 degrees of freedom, however, unlike translational velocities, inertial angular velocities cannot be defined as simply the rates of change of their corresponding motions. More specifically, angular velocities are not the rates of change of Euler angles because the rotations that define each Euler angle originate from different reference frames. In other words, although ψ was rotated around the global Z axis, the second rotation ɵ was rotated about an intermediate y transitional axis, and the third and final rotation φ was rotated about an x axis with reference to the final or local frame. Therefore, each angular velocity is defined by components that are cast in the final reference frame and the sum of these components equals the total angular velocity. The following equation defines global angular velocity components $[\dot{\varphi} ; \dot{\theta} ; \dot{\psi}]$ in terms of local angular velocity components [p ; q ; r].

Note that local angular velocities p, q, and r are measured using onboard rate gyros (Healey, 2001).

- Roll rate (p)
- Pitch rate (q)
- Yaw rate (r)

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix}=\begin{bmatrix}1 & \sin\phi\tan\theta & \cos\phi\tan\theta\\ 0 & \cos\phi & -\sin\phi\\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta\end{bmatrix}\begin{bmatrix}p\\ q\\ r\end{bmatrix}$$

D. EQUATIONS OF MOTION

Now that kinematics has been addressed, we can examine vehicle dynamics. Professor Healey, in Chapter II of his notes entitled Dynamics and Control of Mobile Robotic Vehicles, provides 6 equations of motion. Each of the subsequent 6 equations is derived from either vehicle translation or vehicle rotation. Regarding vehicle translational motion we have equations for surge, sway, and heave that account for all the applied forces felt by the submerged vehicle including gravity and buoyancy along with the forces resulting from added mass. The last 3 equations for roll, pitch, and yaw are derived from vehicle rotation and account for all the applied moments about the vehicle's center line or axis of symmetry.

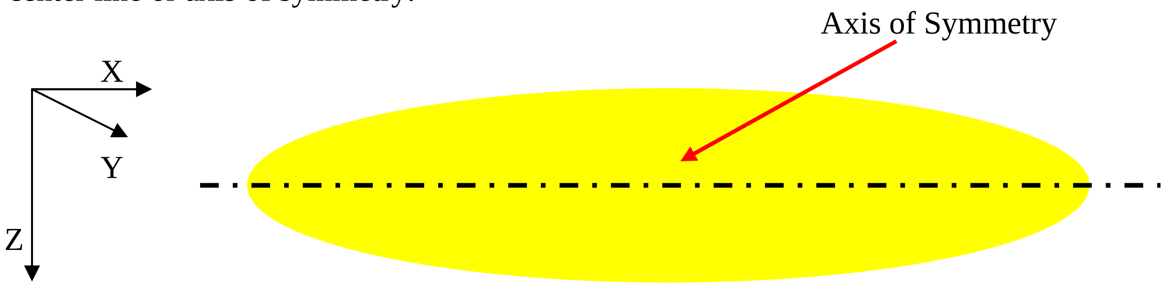
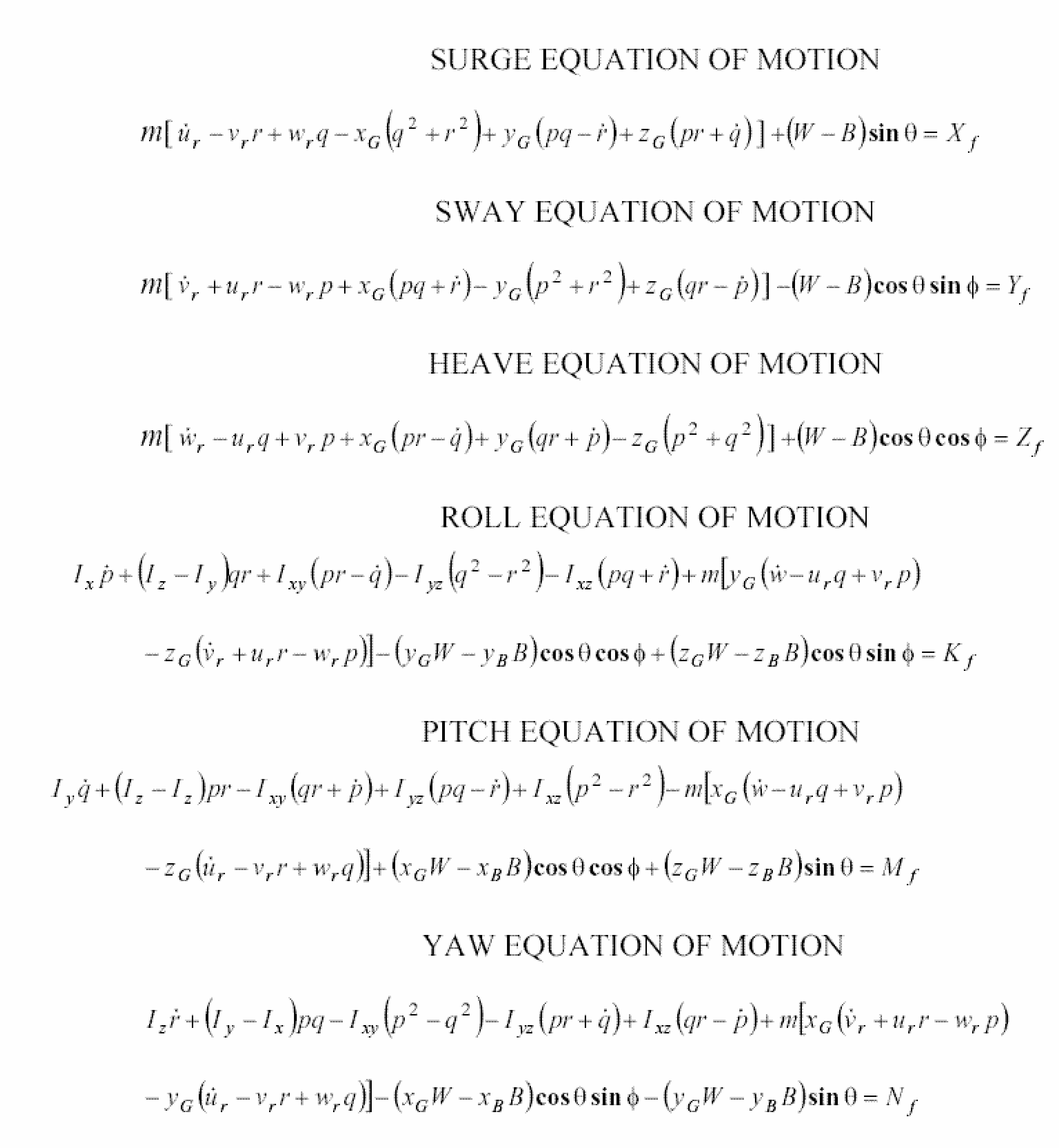


Figure 4. Axis of Symmetry

.



SURGE EQUATION OF MOTION

$$m[\,\dot{u}_r - v_r r + w_r q - x_G\left(q^2 + r^2\right) + y_G\left(pq - \dot{r}\right) + z_G\left(pr + \dot{q}\right)\,] + \left(W - B\right)\mathbf{sin}\,\theta = X_f$$

SWAY EQUATION OF MOTION

$$m[\,\dot{v}_r + u_r r - w_r p + x_G\left(pq + \dot{r}\right) - y_G\left(p^2 + r^2\right) + z_G\left(qr - \dot{p}\right)\,] - \left(W - B\right)\mathbf{cos}\,\theta\,\mathbf{sin}\,\phi = Y_f$$

HEAVE EQUATION OF MOTION

$$m[\,\dot{w}_r - u_r q + v_r p + x_G\left(pr - \dot{q}\right) + y_G\left(qr + \dot{p}\right) - z_G\left(p^2 + q^2\right)\,] + \left(W - B\right)\mathbf{cos}\,\theta\,\mathbf{cos}\,\phi = Z_f$$

ROLL EQUATION OF MOTION

$$I_x\dot{p} + \left(I_z - I_y\right)qr + I_{xy}\left(pr - \dot{q}\right) - I_{yz}\left(q^2 - r^2\right) - I_{xz}\left(pq + \dot{r}\right) + m\left[y_G\left(\dot{w} - u_r q + v_r p\right)\right.$$

$$\left. - z_G\left(\dot{v}_r + u_r r - w_r p\right)\right] - \left(y_G W - y_B B\right)\mathbf{cos}\,\theta\,\mathbf{cos}\,\phi + \left(z_G W - z_B B\right)\mathbf{cos}\,\theta\,\mathbf{sin}\,\phi = K_f$$

PITCH EQUATION OF MOTION

$$I_y\dot{q} + \left(I_z - I_z\right)pr - I_{xy}\left(qr + \dot{p}\right) + I_{yz}\left(pq - \dot{r}\right) + I_{xz}\left(p^2 - r^2\right) - m\left[x_G\left(\dot{w} - u_r q + v_r p\right)\right.$$

$$\left. - z_G\left(\dot{u}_r - v_r r + w_r q\right)\right] + \left(x_G W - x_B B\right)\mathbf{cos}\,\theta\,\mathbf{cos}\,\phi + \left(z_G W - z_B B\right)\mathbf{sin}\,\theta = M_f$$

YAW EQUATION OF MOTION

$$I_z\dot{r} + \left(I_y - I_x\right)pq - I_{xy}\left(p^2 - q^2\right) - I_{yz}\left(pr + \dot{q}\right) + I_{xz}\left(qr - \dot{p}\right) + m\left[x_G\left(\dot{v}_r + u_r r - w_r p\right)\right.$$

$$\left. - y_G\left(\dot{u}_r - v_r r + w_r q\right)\right] - \left(x_G W - x_B B\right)\mathbf{cos}\,\theta\,\mathbf{sin}\,\phi - \left(y_G W - y_B B\right)\mathbf{sin}\,\theta = N_f$$

Where:

W = weight
B = Buoyancy
I = mass moment of inertia terms
ur, vr, wr = component velocities for a body fixed system with respect to the water
p, q, r = component angular velocities for a body fixed system
xB, yB, zB = position difference between geometric center and center of buoyancy
xG, yG, zG = position difference between geometric center and center of gravity

Xf, Yf, Zf, KF, Mf, Nf = sums of all external forces acting in the particular body fixed direction

E. VERTICAL PLANE SIMPLIFICATIONS

Since this thesis deals with vertical plane motion only, we can ignore equations for surge, sway, roll, and yaw and concentrate on variables…

Observed states

- w_r, heave velocity
- q, pitch rate
- ɵ, pitch
- Z, depth

Assuming constant forward speed, the following set of equations models the diving system response to control surface deflections.

$$\boxed{\begin{array}{l} \boldsymbol{x}(\mathrm{t}) = [\mathrm{w(t), q(t)}, \boldsymbol{\theta}(\mathrm{t}), \mathrm{Z(t)}]'; \\ \dot{x}(t) = \boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{M}^{-1}(\boldsymbol{B}\delta_s(t) + \boldsymbol{E}); \end{array}}$$

Where x(t) is a state matrix and $\dot{\mathbf{x}}(t)$ models the vehicle's response in the vertical plane with respect to time. Within $\dot{\mathbf{x}}(t)$, the input matrices A and B contain hydrodynamic coefficients that depend on the geometry of the vehicle being modeled. The hydrodynamic coefficients used for this work were taken from a previous thesis by Prestero in 2001. M is defined as a mass matrix while the matrix E accounts for the mismatch between a submersible's weight and buoyancy. $\delta_s(t)$ is the time varying command for stern plane deflection.

$$\boldsymbol{M} = \begin{bmatrix} (m - Z_{\dot{w}_r}) & -Z_{\dot{q}} & 0 & 0 \\ -M_{\dot{w}_r} & (I_{yy} - M_{\dot{q}}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \; \boldsymbol{A} = \begin{bmatrix} Z_{w_r} & (mU_0 + Z_q) & 0 & 0 \\ M_{w_r} & M_q & \left(z_B B - z_G W\right) & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & U_0 & 0 \end{bmatrix}$$

$$\boldsymbol{B} = \begin{bmatrix} Z_\delta \\ M_\delta \\ 0 \\ 0 \end{bmatrix}; \qquad \boldsymbol{E} = \begin{bmatrix} (W - B) \\ 0 \\ 0 \\ U_{cz} \end{bmatrix}$$

THIS PAGE INTENTIONALLY LEFT BLANK

III. CONTROL METHODS

A. SLIDING MODE CONTROLLER / FEEDFORWARD PREVIEW CONTROLLER

Land robotic vehicles enjoy the luxury of operating on solid ground. Unfortunately, underwater robotic vehicles do not enjoy the same luxury. Suspended underwater, marine vehicles are forced to constantly adjust and readjust to stabilize themselves against ocean wind, waves, and currents. For the last several decades, manufacturers of AUV systems have employed control systems called autopilots, which "stabilize the motion of the vehicle in response to steady commands" (Healey, 2001). Autopilots rely on feedback collected from onboard sensors to supply commands to vehicle actuators, which control the motion of the vehicle (Healey, 2001).

As mentioned earlier, REMUS is smaller and less cumbersome than other models of AUVs, however, this translates into problems concerning vehicle stability. With more maneuverability also comes higher expectations and missions that require operating in tightly confined underwater spaces. With less room to operate, a very small margin of error exists; therefore a reliable and responsive controller is needed to assure the vehicle's hydrodynamic stability is not compromised. With regards to obstacle avoidance, quickly responding to changes in the ocean environment is crucial.

For this thesis, in order to maintain a cruising altitude of 3 meters, REMUS relied on a sliding mode controller autopilot to ensure stability while navigating. This choice was made given the following reasoning (Healey 2001).

- SMC compensates for known nonlinear behavior
- SMC provides robustness to uncertainty
- SMC is easy to use

The term nonlinear refers to the fact that the system we wish to model, in our case an underwater environment, cannot be expressed through first order equations. The modeler must therefore rely on second or higher order equations to simplify an unpredictable environment. Fortunately, sliding mode control reduces the order of systems making them easier to model.

Prior to implementing a sliding mode controller, a sliding surface (σ) must be defined. This surface is a scalar function resulting from a linear combination of state variables such as position, velocity, acceleration etc. The aim is to drive the system to the sliding surface and ultimately to the condition σ = 0 while making sure the state variables are always reducing (Healey, 2001).

Suppose we wish to model a second order nonlinear system. The corresponding sliding surface would be defined as

$\sigma = \lambda x_1 + x_2$, Where λ is an unknown frequency (radians/sec).

Using Lyapunov methods, the control law (u) is subsequently formulated by defining a positive definite function, V(σ) > 0 where the derivative of this function for all times greater than zero is negative (Healey, 2001). By defining a positive definite Lyapunov function's derivative as negative, we guarantee that our sliding surface (σ) is always reducing.

$$u = \{b(x_2) + kx_1\} - \{\lambda x_2\} - \eta sat\,\mathrm{sgn}(\sigma / \phi)$$

Equipped with a simulated combination of RDI Doppler and forward look blazed array sonar, REMUS was capable of continuously observing its own altitude while also collecting range and bearing sonar data of bottom contours out to 60 meters. Equipped with the sliding mode controller autopilot used for maintaining depth, REMUS also relied on a feedforward preview controller. Processed forward look sonar images were used to determine the magnitudes of certain parameters within a Gaussian potential function. With the origin of the potential function placed at the exact position of the previewed obstacle, an altered trajectory was commanded and REMUS was able to subsequently avoid a collision. In other words, REMUS' feedforward preview controller was able to adjust the vehicle's flight path in order to avoid an oncoming obstacle relying solely upon processed forward look sonar images.

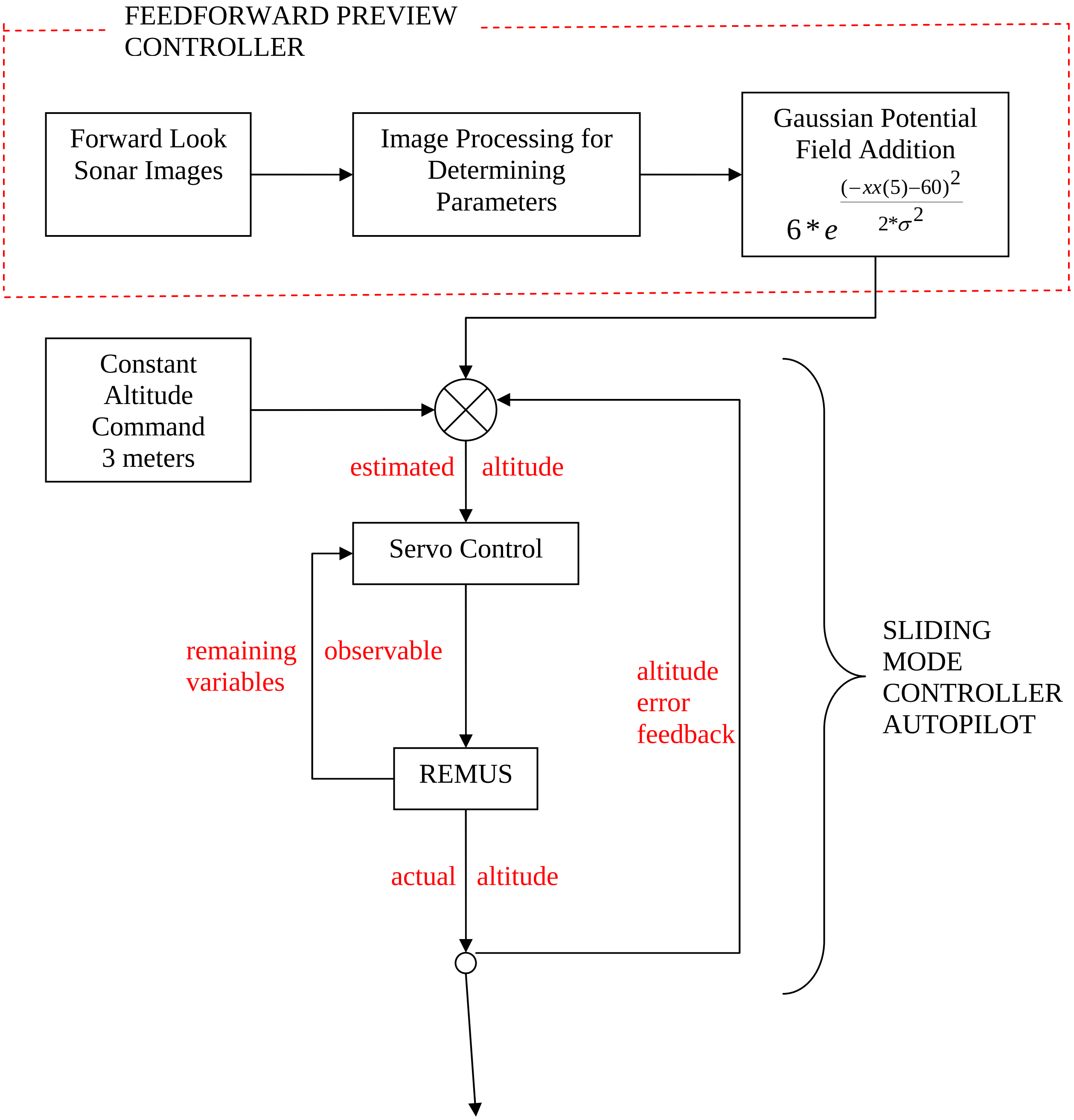

FEEDFORWARD PREVIEW CONTROLLER
Forward Look Sonar Images
Image Processing for Determining Parameters
Gaussian Potential Field Addition
$6*e^{\frac{(-xx(5)-60)^2}{2*\sigma^2}}$
Constant Altitude Command 3 meters
estimated altitude
Servo Control
remaining variables
observable
altitude error feedback
REMUS
actual altitude
SLIDING MODE CONTROLLER AUTOPILOT

Figure 3. Block Diagram of Combination of Feedforward Preview Controller and Sliding Mode Altitude Controller

Figure 3 depicts REMUS' entire obstacle avoidance procedure beginning with obstacle detection through image processing and ending with REMUS actually avoiding the obstacle. Obstacle range and bearing data is extracted form sonar images received from the blazed array forward look sonar. These sonar images define the dimensions of the obstacle and as a result, provide optimized magnitudes of certain variables within the following Gaussian potential function.

$$6 * e^{\frac{(-xx(5)-60)^2}{2*\sigma^2}}$$

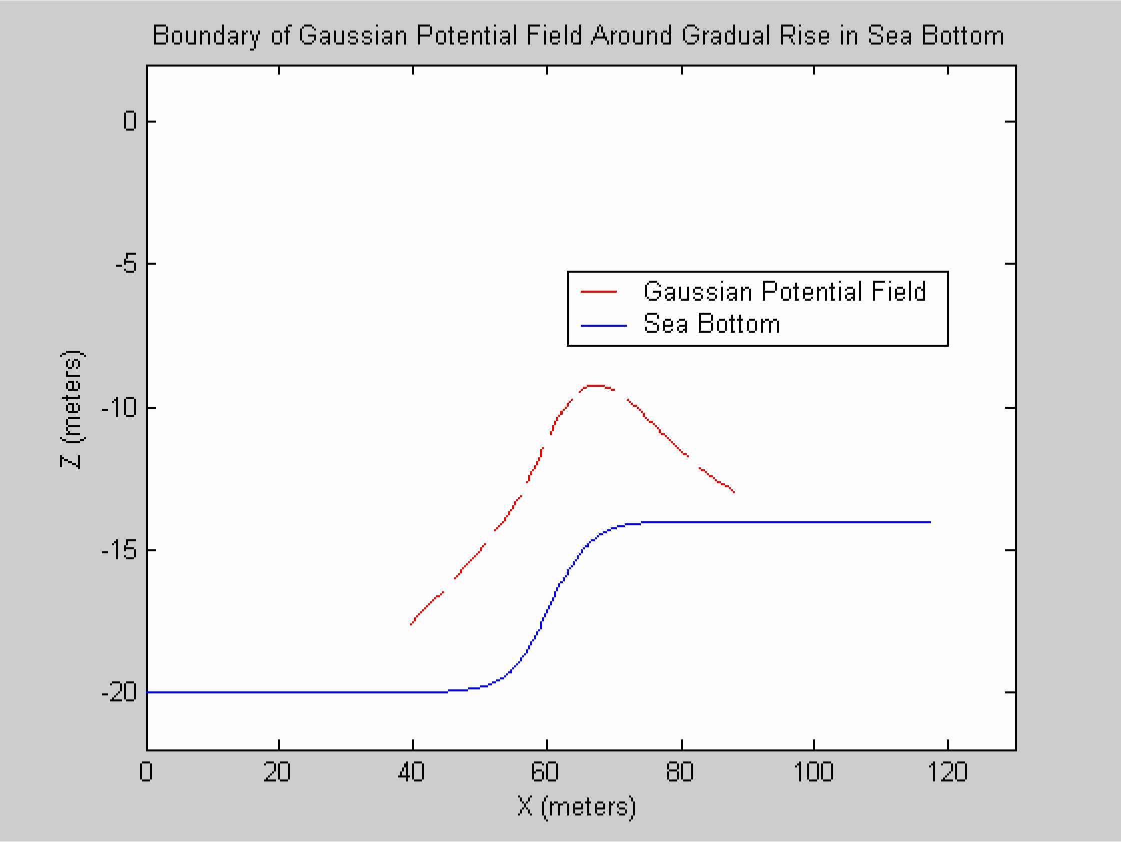


Figure 4. Gaussian Potential Field Boundary Around Sea Bottom Rise

As seen in Figure 4, the Gaussian potential function essentially defines REMUS' trajectory through a field made up of both attractive and repulsive forces. Addressing the autonomous navigation problem with potential fields makes sense for the following reasons. Not only is the obstacle avoidance path smooth and efficient, but the vehicle's trajectory does not need to be planned in advance. The characteristics of the potential function alone control vehicle avoidance maneuvers by creating a repulsive field around an obstacle that forces the vehicle to trace the potential field in order to regain it's commanded trajectory. Also, potential fields are flexible and can be updated in real time using the feedforward preview controller to assign optimized magnitudes to certain parameters within the actual Gaussian potential function.

To fully optimize REMUS' obstacle avoidance capability, each variable's contribution within the Gaussian potential function must be examined. Beginning with the scaling factor, the value of 6 represents the overall height of the potential function. The magnitude of this number in essence bounds the vehicle's adjusted trajectory. Therefore, depending on the oncoming obstacle's height, this parameter may be adjusted accordingly. The numerator of the exponent defines the location of the potential function along the sea bottom. Since the forward look sonar images provide us with the oncoming obstacle's range, we are able to subtract the obstacle's global x position, 60, from xx(5), the distance REMUS has traveled in the x direction with respect to the global reference frame. Finally, the parameter σ, defines the width of the potential function. REMUS' initial response depends on the value of this parameter; higher values of sigma force REMUS' obstacle avoidance maneuvers to begin sooner while smaller values of sigma delay evasive maneuvers.

Once the feedforward preview controller has provided a suitable Gaussian potential function, this estimated trajectory is summed with the previously commanded altitude of 3 meters to produce the following estimated attitude command.

$$\text{altcom} = 3 + 6 * e^{\frac{(-xx(5)-60)^2}{2*\sigma^2}}$$

After a suitable avoidance trajectory has been commanded, REMUS' servo control mechanisms adjust the vehicle's control surfaces accordingly. Since REMUS' velocity is constant at 1.5 m/s, the term "adjust" refers to the deflection of the vehicle's stern planes to either pitch the vehicle up or down as needed. Also, throughout this portion of the obstacle avoidance procedure, other observable variables such as heave velocity, pitch rate, and pitch angle are constantly being fed back to the vehicle's servo control mechanisms. Finally, after maneuvering to the commanded estimated altitude, REMUS records its actual altitude and feeds any difference between the estimated altitude and actual attitude back to the original altitude command starting the entire process over again.

In summary, the integration of feedforward preview control and sliding mode control results in an obstacle avoidance controller that is not only robust, but also autonomous.

IV. SPACE SIMULATION

A. 2-D OCEAN ENVIRONMENT MODEL

Before attempting to model REMUS' ability to avoid certain obstacles, we must first define a two dimensional domain consisting of an X directional space and a Z directional space.. Regarding ocean depth, Z values started at a value of zero and increased to a maximum depth of 20 meters. The X direction however, remained unbounded, constrained only by a time window of 80 seconds. Of the four types of ocean floors modeled, the gradual rise and step sea bottoms originated from the MATLAB file entitled seabottom.m, while the hill and wall sea bottom profiles originated from file entitled seabottom_hill.m.

1. Sea Floor Model: Gradual Rise and Step

Using a series of "if" statements in the MATLAB file seabottom.m, the sea bottom slope was manipulated using the following hyperbolic tangent function.

$$\begin{aligned}
&X \le 60.................H = 20\\
&X \ge 60+1...........H = 20-S\\
&60 < X < 60+1....H = 20 - S*0.5*(1+\tanh((X-60-0.5*1)/l)
\end{aligned}$$

Where H is depth measured from the water's surface (Z = 0), S is the amplitude or total rise in meters of the obstacle, and *l* is wavelength. The MATLAB file remusderivalt.m referenced the above function simulating either a gradual rise or sharp step depending on the chosen wavelength. In general, the sea bottom slope decreased with higher wavelengths. Note that since simulation was run continuously over a time interval of 80 seconds, for each time step there existed a value for H along with a corresponding value for S. Therefore, at any point in time within that 80 second time span, the sum of any corresponding H and S value should equal 20, the depth boundary in meters.

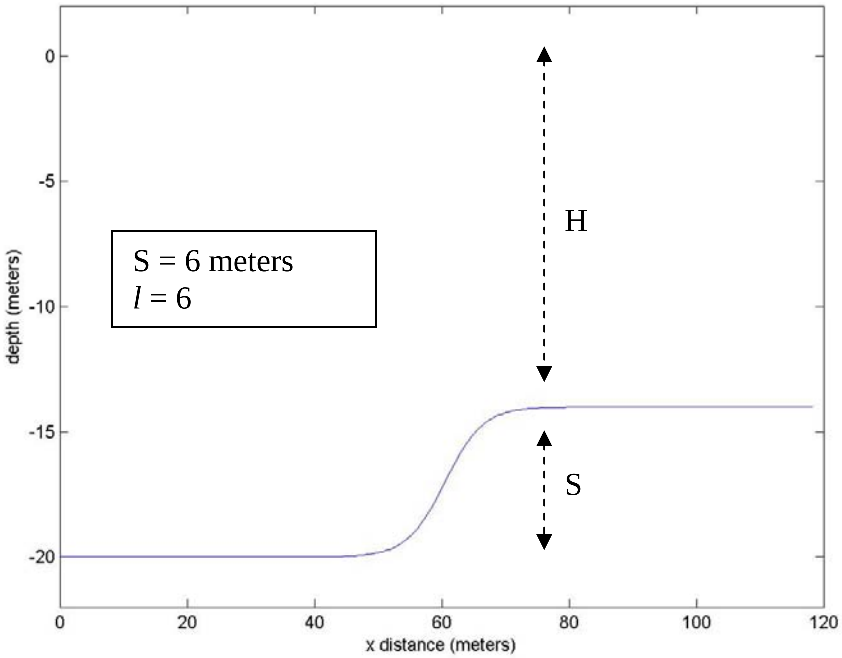


Figure 5. Simulated Gradual Rise Sea Bottom

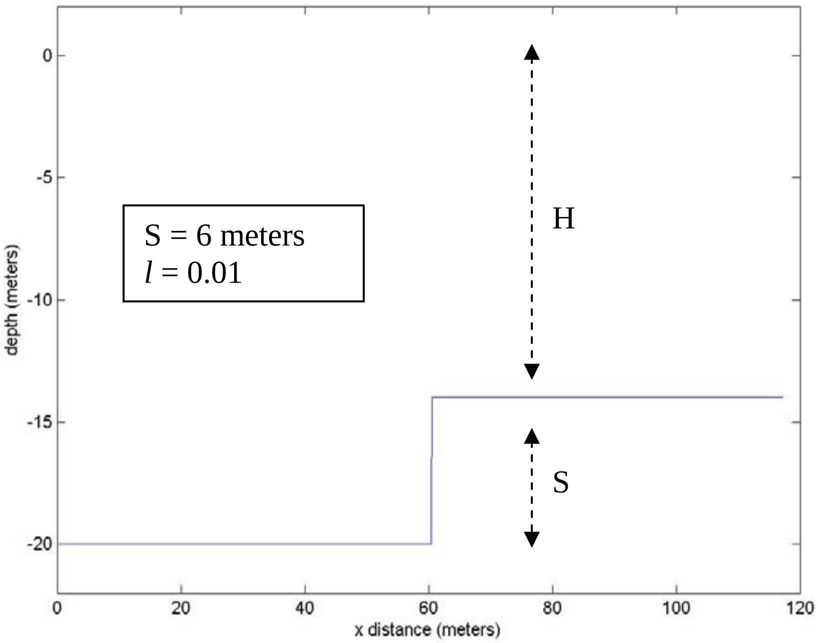


Figure 6. Simulated Sharp Step Sea Bottom

2. Sea Floor Model: Hill and Wall

Similar to the gradual rise and step sea bottoms, the hill and wall bottom contours resulted from an exponential function embedded in a MATLAB file named seabottom_hill.m.

$$H = 20 - S * \exp(-(X - 60).\hat{}2/2/l)$$

Where the height and width of the hill are controlled through modifications made to variables S and *l* respectively. In general, width increased as wavelength increased.

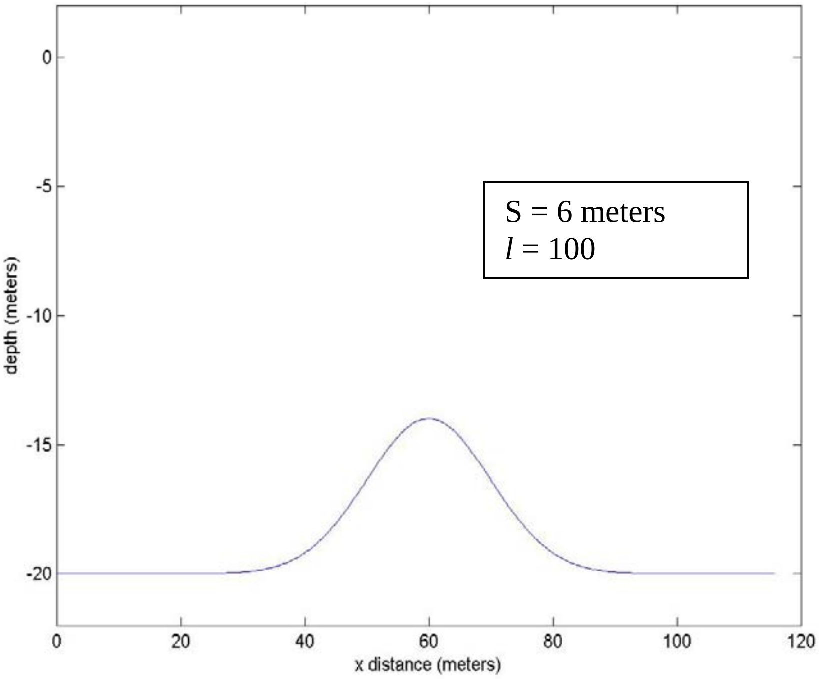


Figure 7. Simulated Hill Sea Bottom

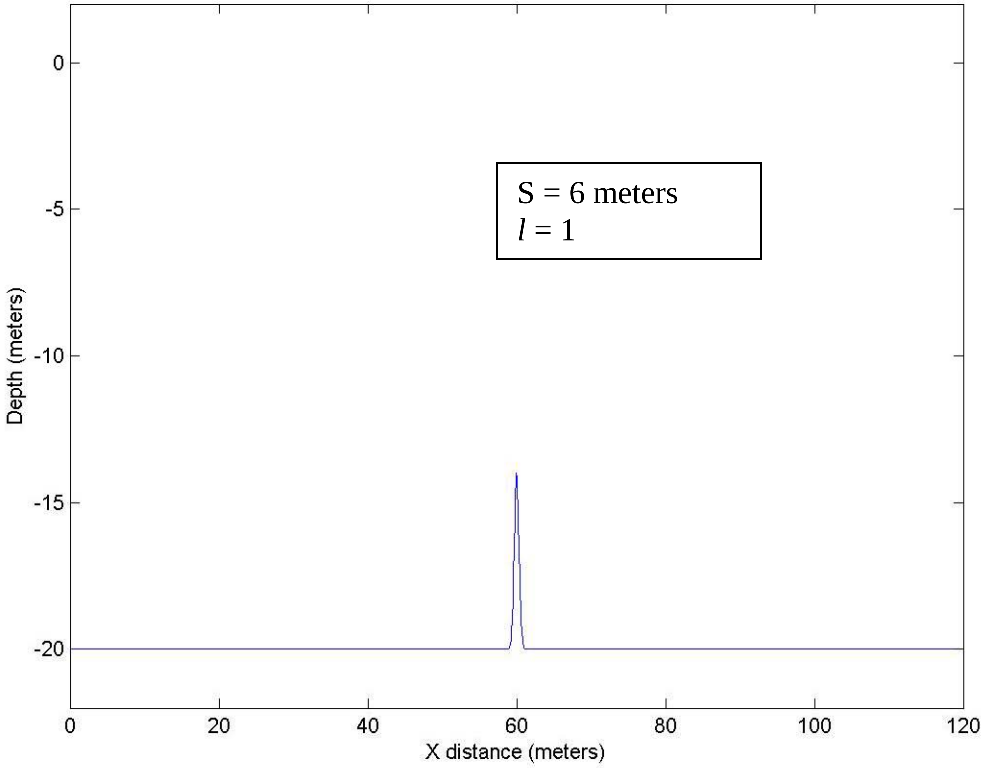


Figure 8. Simulated Seawall

B. STATE INTEGRATION

During simulation, REMUS was commanded to maintain a cruising altitude of 3 meters above sea bottom. Throughout the 80 second simulation, four observable states pertaining to vertical plane motion were integrated continuously using the vertical plane equations of motion along with the ode45 function embedded in the MATLAB file REMUSCHRIS.m created by Chris Churan in May 2003. This resulted in five different arrays containing state values for each individual time step that were stored in the MATLAB workspace. The five observable states included

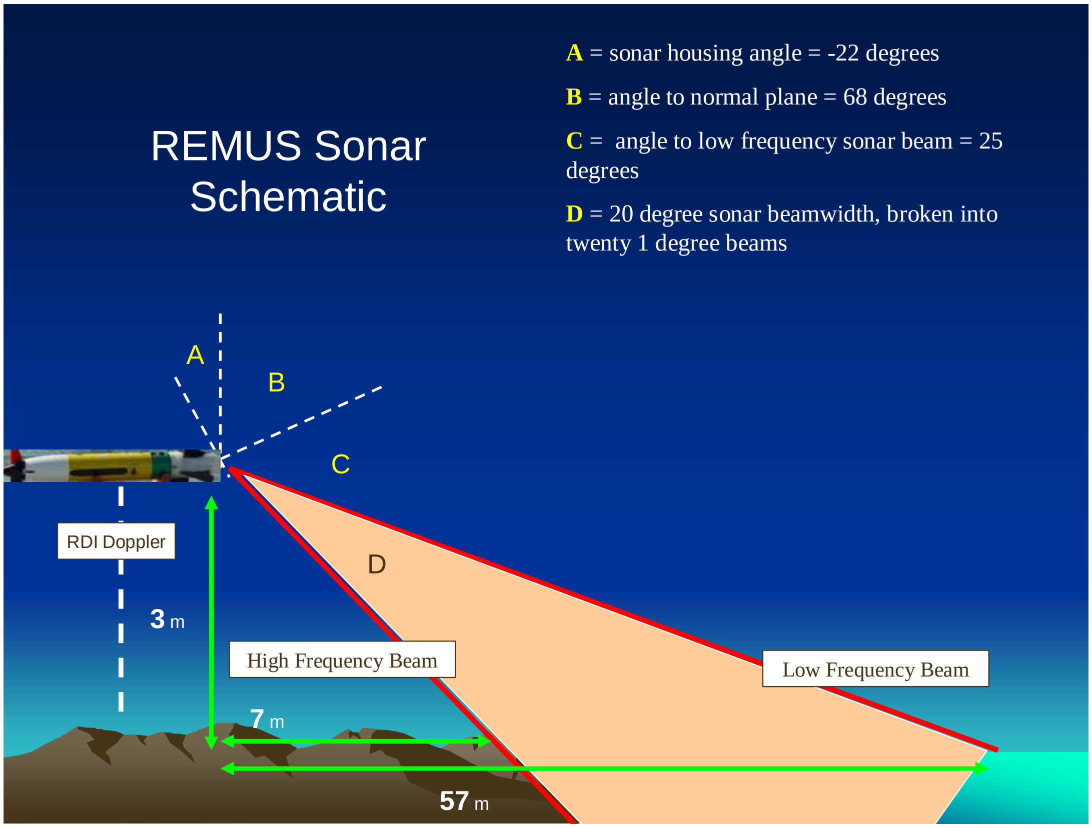
- x(1) = q, pitch rate
- x(2) = w, heave velocity
- x(3) = ɵ, pitch angle

- x(4) = z, depth of REMUS

- xx(5) = X, distance from global origin along x axis

C. SONAR MODEL

REMUS was equipped with two sensors; an RDI Doppler and a forward look sonar. While the RDI Doppler was used to maintain the altitude command of 3 meters, the forward look sonar acted as the primary obstacle avoidance sensor.



REMUS Sonar Schematic

A = sonar housing angle = -22 degrees

B = angle to normal plane = 68 degrees

C = angle to low frequency sonar beam = 25 degrees

D = 20 degree sonar beamwidth, broken into twenty 1 degree beams

Figure 9. REMUS Sonar Schematic: RDI Doppler and Forward Look Sonar

Figure 9 specifies the range and alignment of both the RDI Doppler and forward look sonar. The RDI Doppler was located on the forward underside of REMUS and simply ensured that the vehicle maintained the altitude command of 3 meters. In order to accommodate a reasonable forward look sonar range, the sonar housing staves were tilted back 22 degrees. This allowed for a maximum sonar range of roughly 60 meters, enough to comfortably detect obstacles in REMUS' path. The actual forward look sonar consisted of two 10 degree halves; one high frequency half and one low frequency half. Each individual sonar beam produced a return consisting of both range to and bearing of its reflection point. The following figure illustrates the mechanics behind REMUS' simulated forward look sonar defining each reflected range and bearing using fundamental mathematical properties.

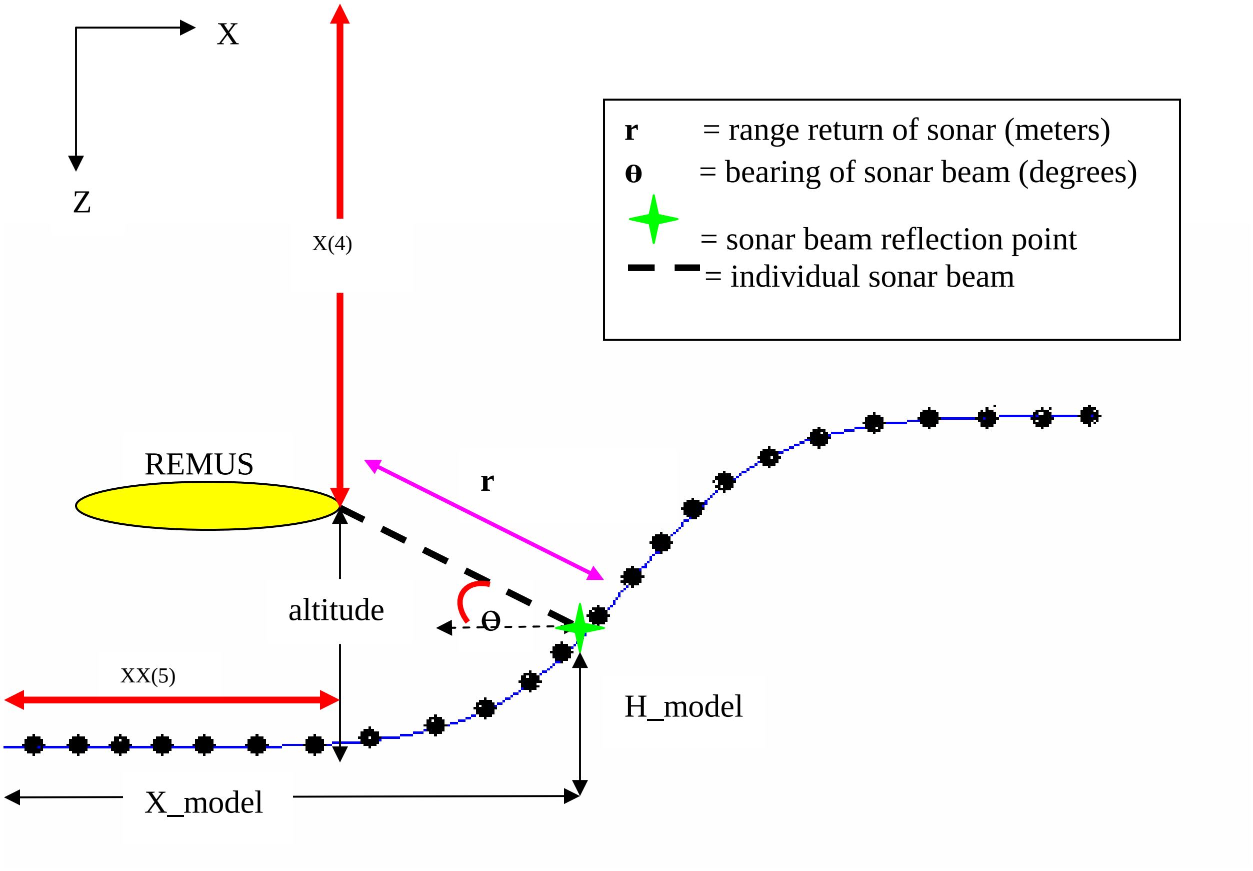

X
Z
X(4)
r = range return of sonar (meters)
θ = bearing of sonar beam (degrees)
= sonar beam reflection point
= individual sonar beam
REMUS
r
altitude
θ
XX(5)
H_model
X_model

Figure 10. Mechanics Behind REMUS' Simulated Forward Look Sonar

Figure 10 defines range and bearing calculations regarding REMUS' simulated forward look sonar. Range data resulted from the Pythagorean Theorem and more specifically the equation,

$$r = \sqrt{(X\_\mathrm{mod}el - xx(5))^2 + (H\_\mathrm{mod}el - x(4))^2}\,.$$

Similarly, pitch corrected bearing data resulted from the trigonometric property of sine,

$$b = \sin^{-1}\left[\frac{H\_model - x(4)}{r}\right]$$

Throughout simulation, the ocean floor was modeled using a finite number of equally spaced dots or points in XZ space. Each dot was assigned an x distance from the global origin defined as X_model along with an altitude defined as H-model. The variables x(4) and xx(5) represent REMUS' Z and X position in real time respectively. As a result, the simulated forward look sonar records a range and bearing for each individual point along the ocean floor, which change over time according to REMUS' position in XZ space. Finally; to accurately simulate the proper collection beam width of 20 degrees, simple logic code within the MATLAB file remusderivalt.m defined boundaries of collectible bearing and range data. The following figure represents a single snapshot of what REMUS' simulated forward look sonar perceives as the advancing ocean floor.

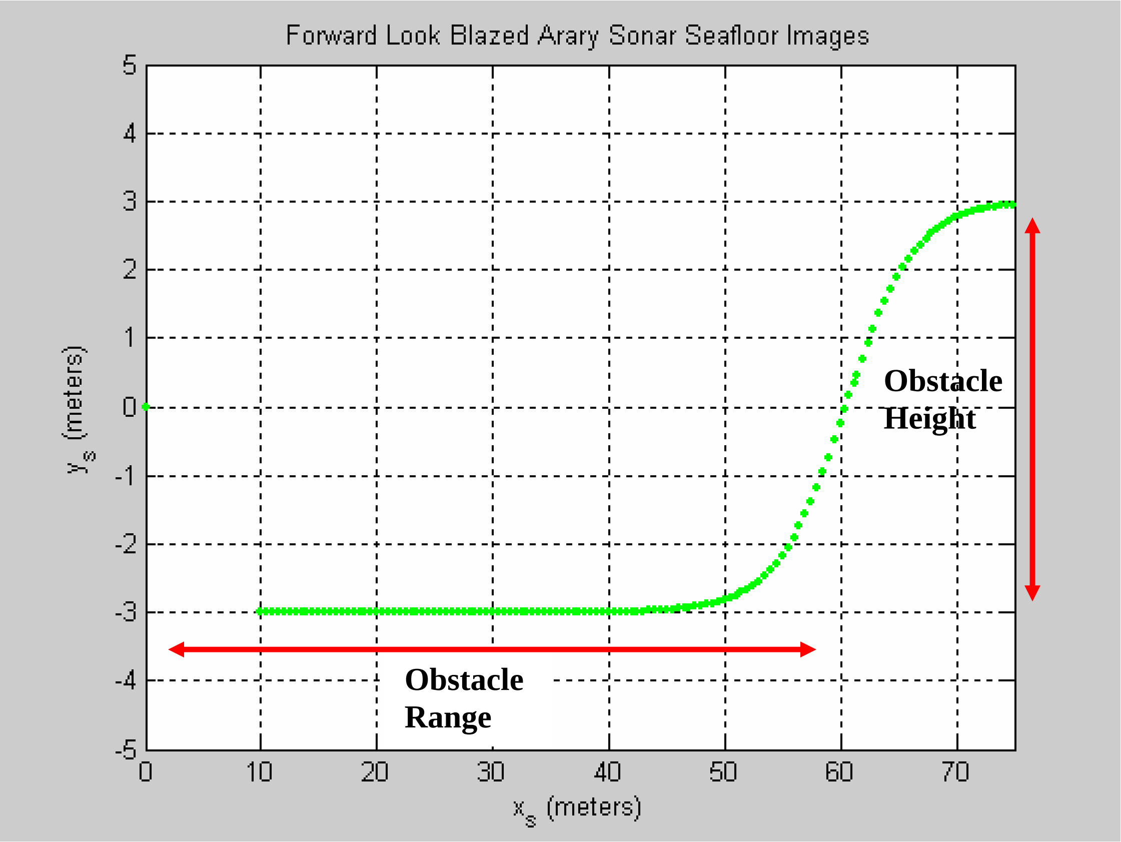


Figure 11. Forward Look Sonar Image of Gradual 6 meter Rise Located at 60 meters Along the Global X axis

From Figure 11, REMUS’ feedforward preview controller gathers all the information required to safely navigate over the obstacle.

More specifically, obstacle range and height according to REMUS’ current position force REMUS’ dynamic controller to command an avoidance trajectory in order to ensure safe navigation over the rising sea bottom.

V. VEHICLE SIMULATION

A. INITIAL TESTS

Initial tests involving simulated REMUS runs over sea bottoms ranging from sharp steps to gradual rises were performed to validate the capabilities of forward look sonar.

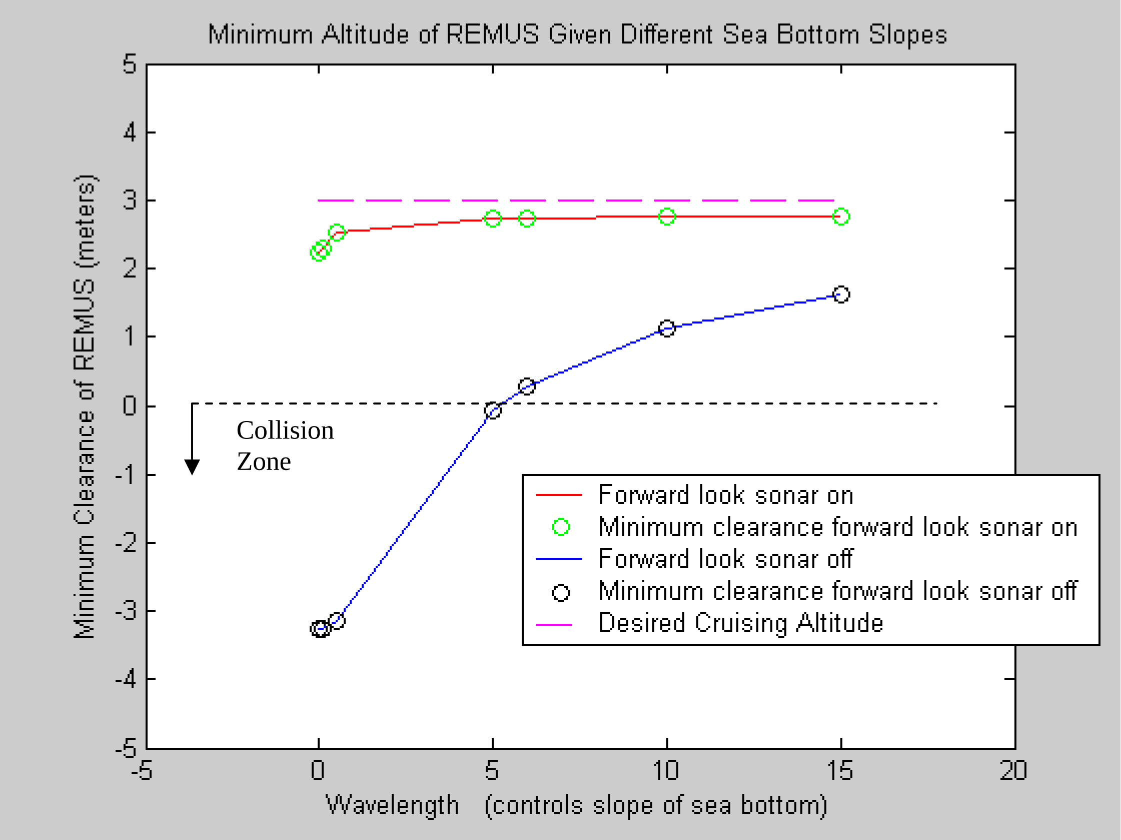


Figure 12. Graph Comparing Minimum Clearance of REMUS With and Without Forward Look Sonar

Figure 12 displays the results of the forward look sonar validation testing. The red line indicates REMUS' cruising altitude over a wide range of sea bottoms while relying on forward look sonar. The blue line indicates REMUS' cruising altitude over identical sea bottoms with no forward look capability. With no forward look sonar, REMUS was forced to maintain a 3 meter commanded altitude using only its RDI

Doppler. As mentioned earlier, attempting to maintain a certain altitude using only RDI Doppler is analogous to driving without being able to see out of your front windshield. Notice that in Figure 12, regardless of the steepness of the encountered sea bottom, REMUS was easily able to avoid collisions with the forward look sonar turned on. On the other hand, with disabled forward look sonar, REMUS was only able to avoid colliding with gradually rising sea bottoms defined by wavelengths of around 5 radians and higher. Figure 13 and 14 clearly illustrate this point. In Figure 13, REMUS' forward look sonar is disabled and therefore cannot react fast enough to adjust to the sloping sea bottom.

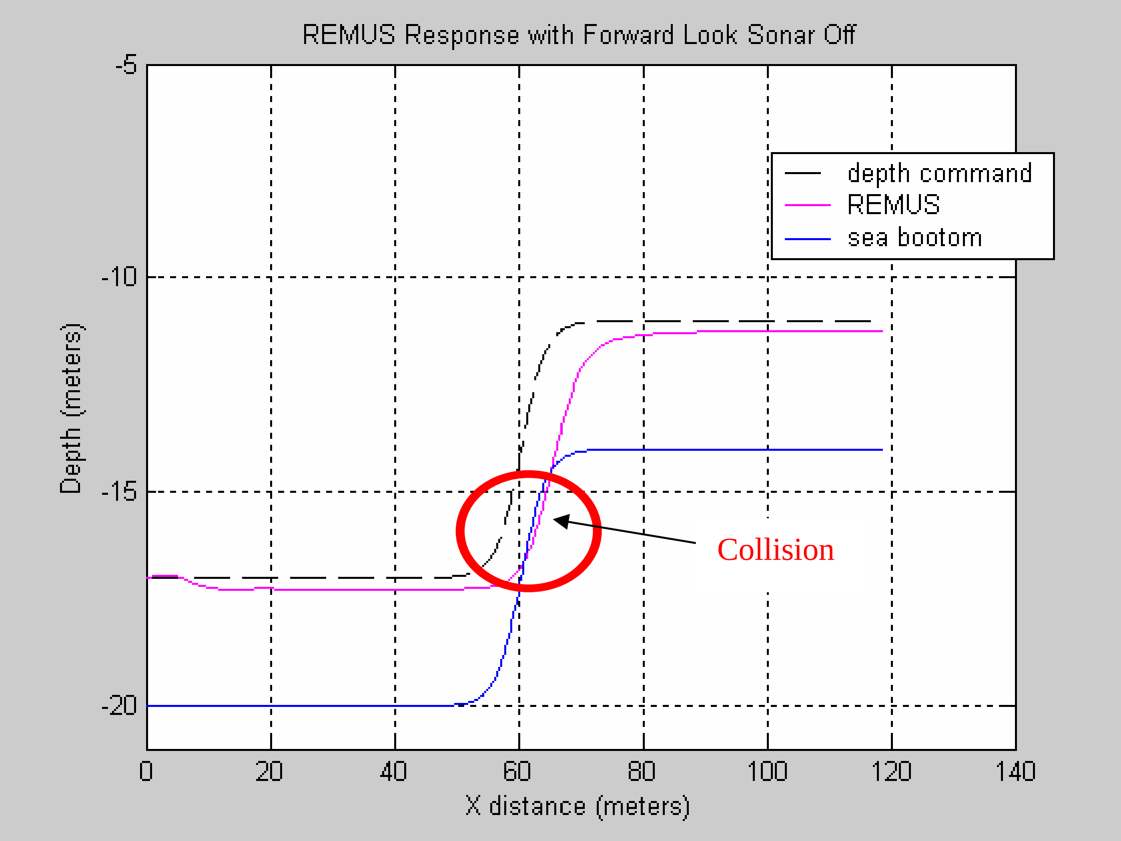


Figure 13. REMUS Collision: Forward Look Sonar Off

Figure 14 displays REMUS' flight path over the exact same sea bottom, only now, forward look sonar is turned on.

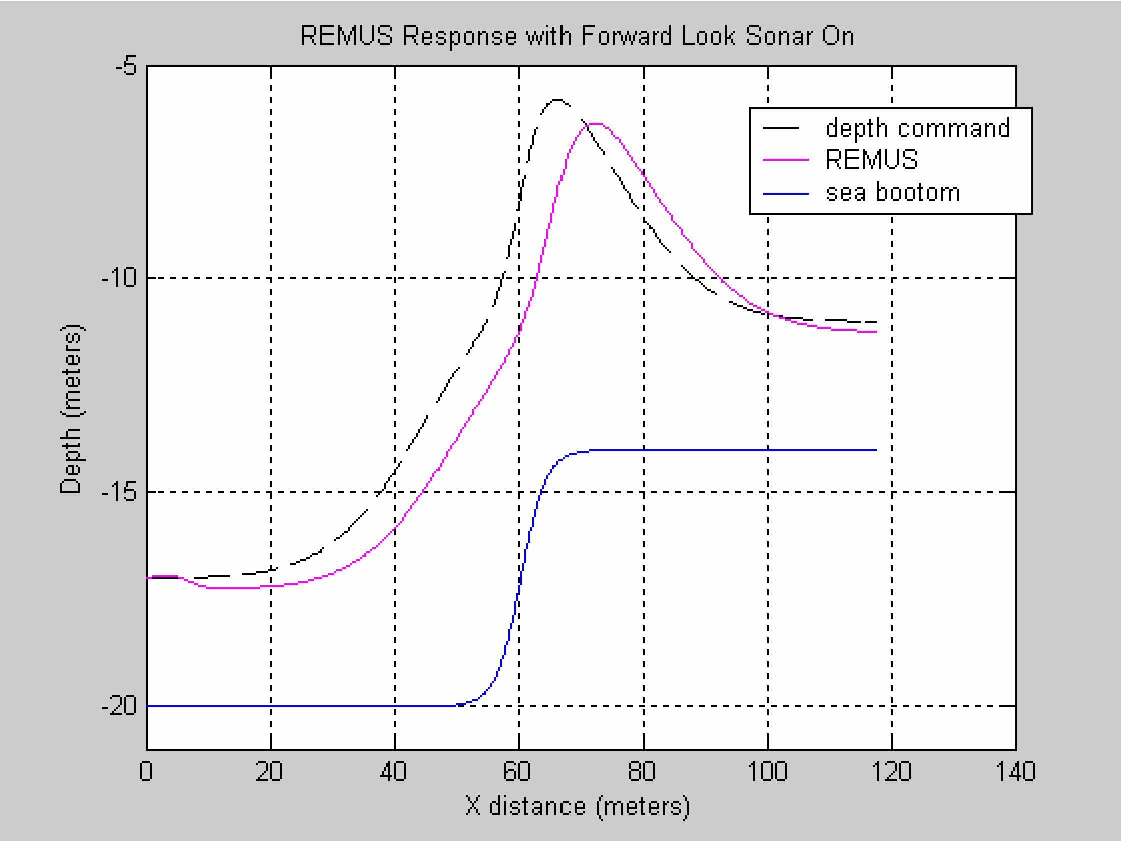


Figure 14. REMUS Obstacle Avoidance: Forward Look Sonar On

In summary, forward look sonar proved essential to safe navigation by enlarging REMUS' field of view and providing advanced warning of approaching obstacles

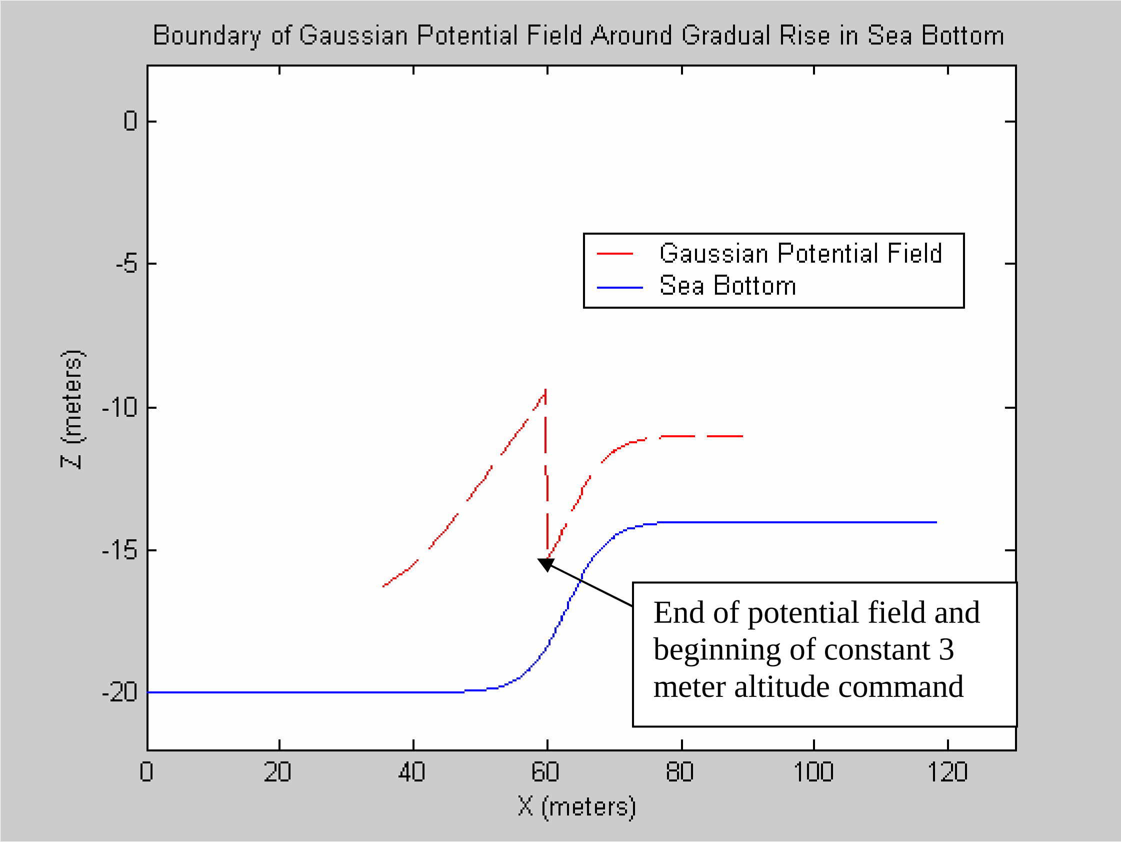
.

B. OBSTACLE AVOIDANCE TESTING

The following simulations model REMUS' flight path over the 4 previously defined sea bottoms; gradual rise, step, hill, and sea wall. During each simulation, REMUS' velocity was held constant at 1.5 m/s and ocean currents were assumed negligible.

1. Optimizing Obstacle Avoidance through Gaussian Potential Field Sizing

In an effort to perhaps ensure a smoother dissent after avoiding an obstacle, REMUS’ initial obstacle avoidance controller utilized only half of a Gaussian potential function. In other words, after sensing an obstacle, REMUS would trace only the first half of the full potential field that was subsequently placed around the obstacle using the feedforward preview controller. Once REMUS had completed navigation around the half potential field, the vehicle’s obstacle avoidance controller immediately began receiving constant altitude commands of 3 meters. Figure 15 displays a typical set of depth commands REMUS would receive when relying on only half a Gaussian potential function while navigating over an obstacle located at 60 meters along the global axis.



End of potential field and beginning of constant 3 meter altitude command

Figure 15. Obstacle Avoidance Depth Commands using Half of a Gaussian Potential Field

The following figures display REMUS' flight paths over a relatively gradual sea bottom slope and a sharp step respectively. Keep in mind that REMUS is using only half of a potential field to navigate over each obstacle.

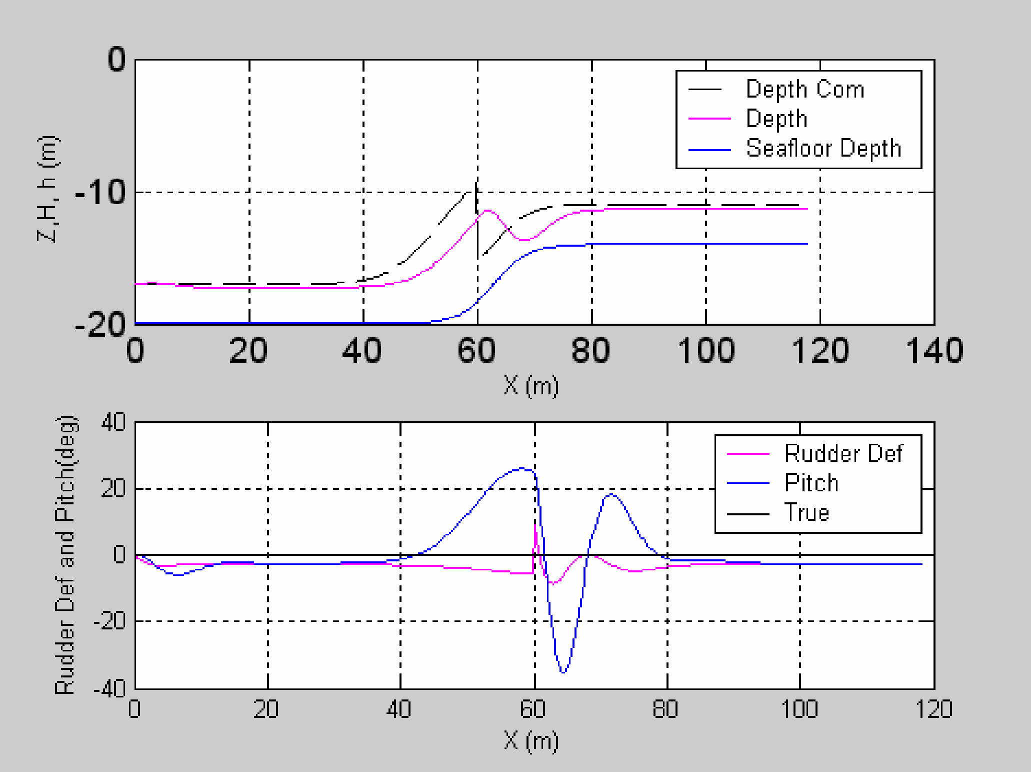


Figure 16. REMUS Dynamic Response over Gradual Rise Using Half of a Potential Field

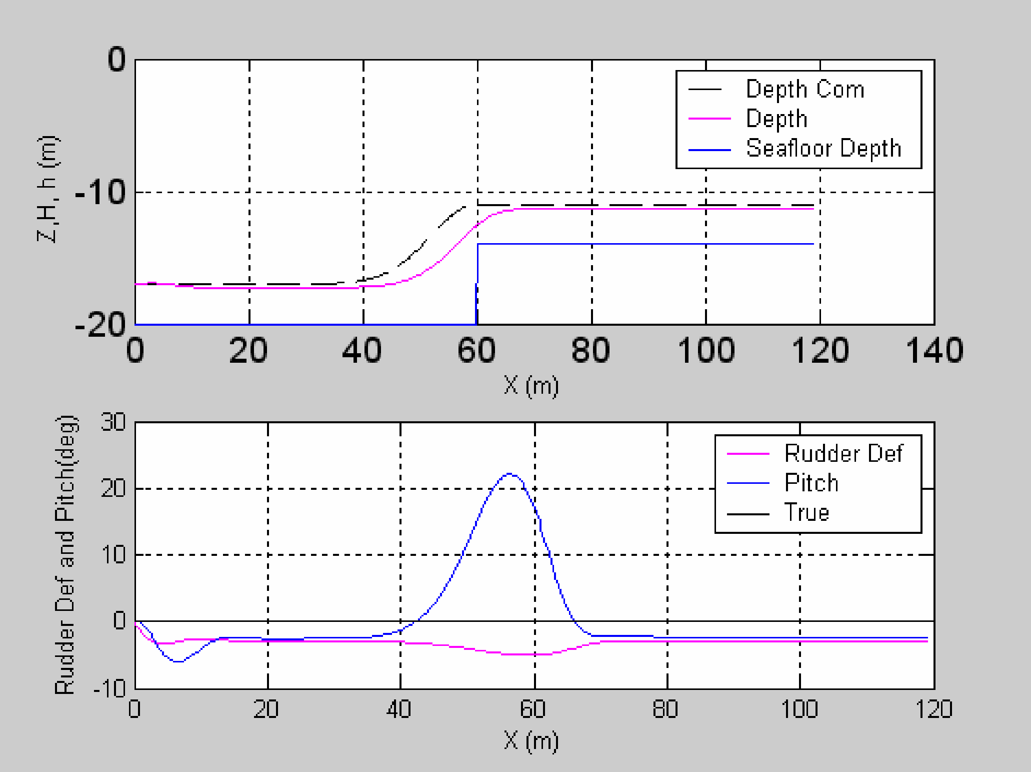


Figure 17. REMUS Dynamic Response over Sharp Step Using Half of a Potential Field

Notice that in Figure 16, REMUS exhibits an oscillatory avoidance response while adjusting to the rising sea bottom. This inefficient flight path results from the vehicle having to relocate the ocean floor where the potential field has been discontinued. Once REMUS relocates the ocean floor, it maneuvers itself relying on RDI Doppler and assumes a cruising altitude of 3 meters. Conversely, as figure 17 illustrates, REMUS responds rather well to a sharp step sea bottom while using the same obstacle avoidance method. Vehicle flight path is much smoother and does not exhibit any oscillatory behavior. However, REMUS passes dangerously close to the sea bottom and although does manage to avoid a collision, a larger minimum clearance is preferable. As discussed earlier, the variable sigma within the Gaussian potential function controls the width of the defined potential field. Therefore, by increasing the magnitude of sigma in accordance with forward look sonar images processed through the feedfoward preview controller, REMUS could achieve a more gradual response. Figure 18 displays this result.

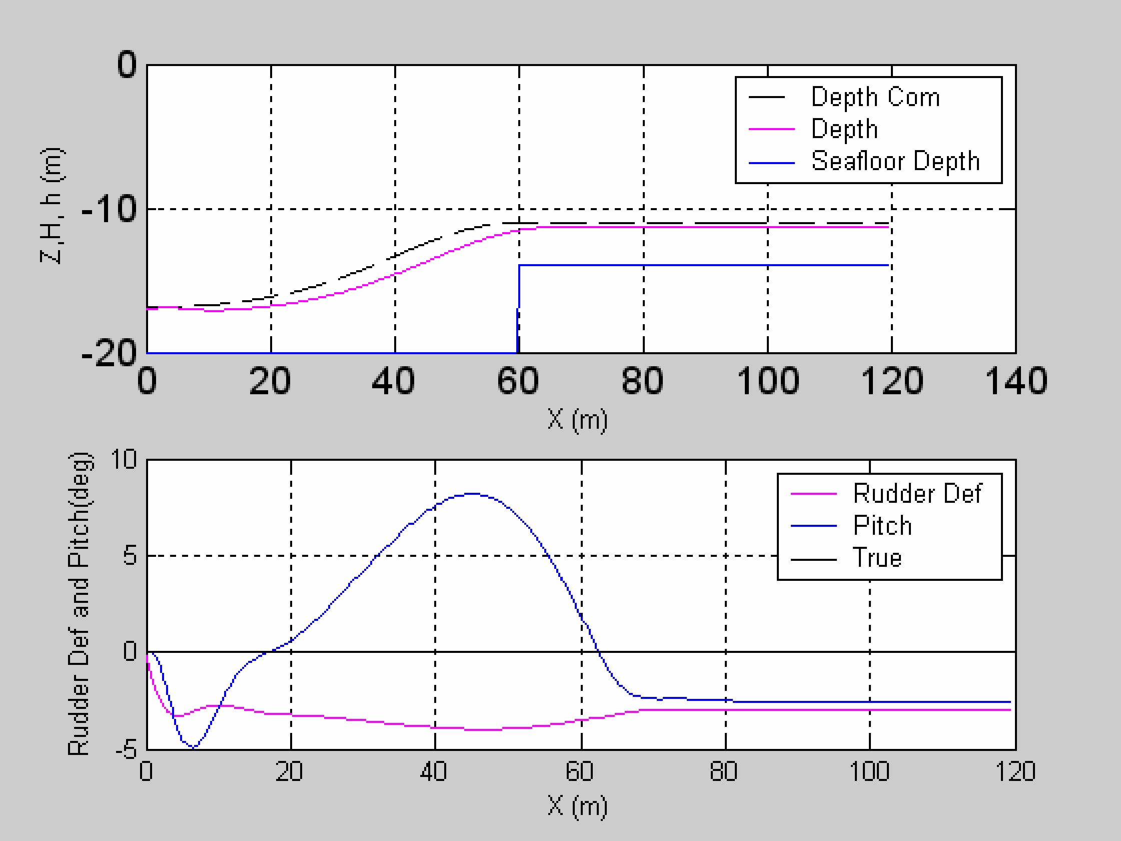


Figure 18. More Gradual Response Achieved through Gaussian Potential Function Variable Manipulation

Nonetheless, regardless of the satisfactory response obtained over a sharp step sea bottom, utilizing a half potential field produced unfavorable results over more gradual rising sea bottoms. As a result, the obstacle avoidance method using only half a potential field was abandoned in favor of the more reliable and capable full potential field approach. The remaining tests therefore involved obstacle avoidance control according to a full Gaussian potential field.

2. REMUS' Dynamic Response Over a 6 Meter Gradual Incline

The following figure displays REMUS' adjusted trajectory in response to a gradual sea floor rise of 6 meters. Notice that unlike the previous response defined by half of a potential field, REMUS' response does not exhibit any rapid pitch oscillations. The following response defined by a full Gaussian potential field is therefore much more efficient, saving the vehicle energy that would have otherwise been spent performing needless pitching motions.

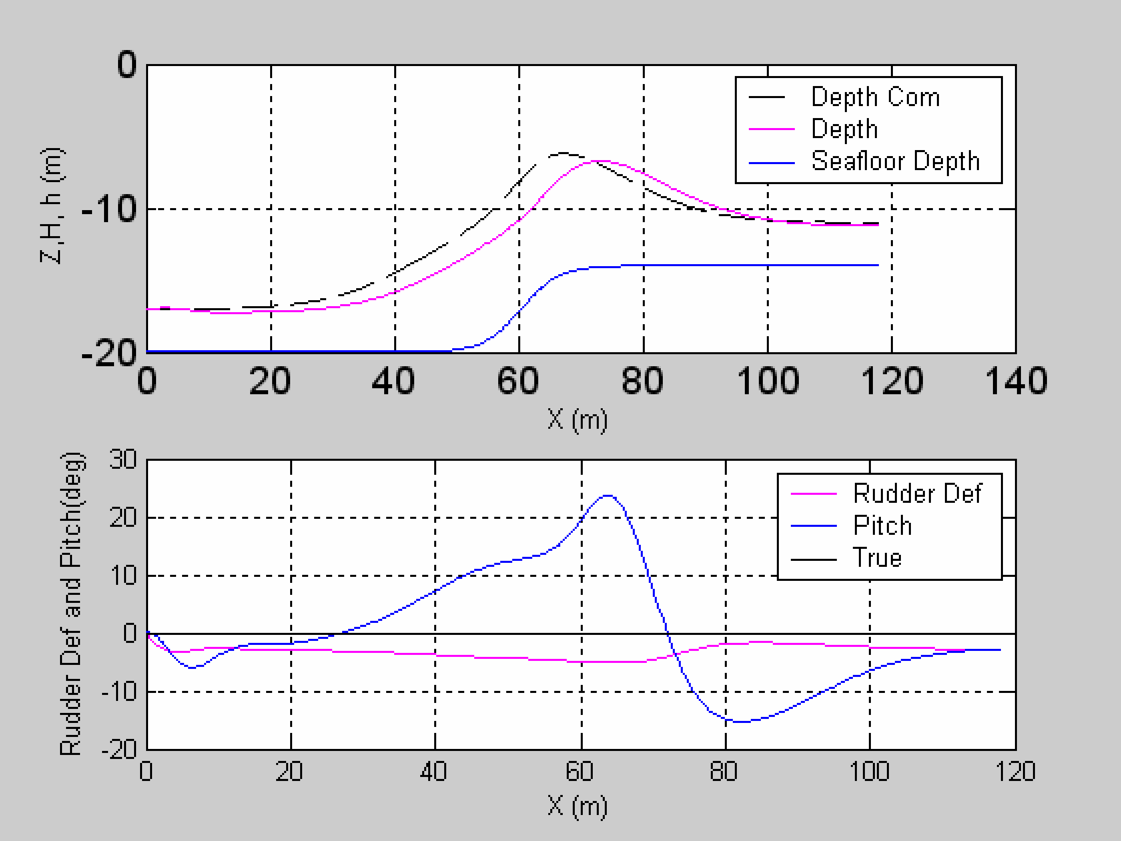
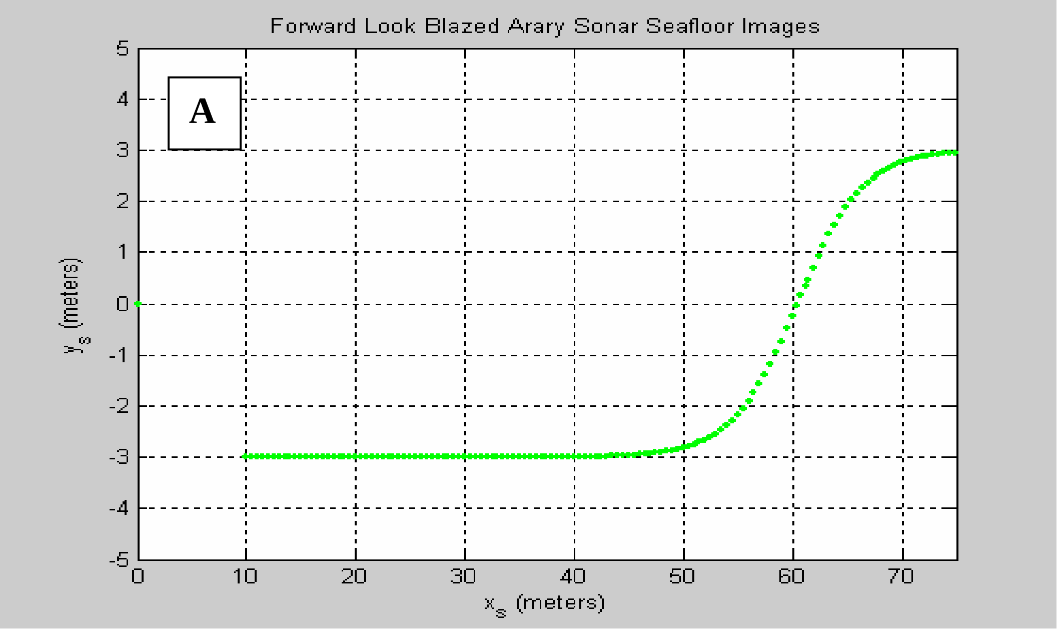
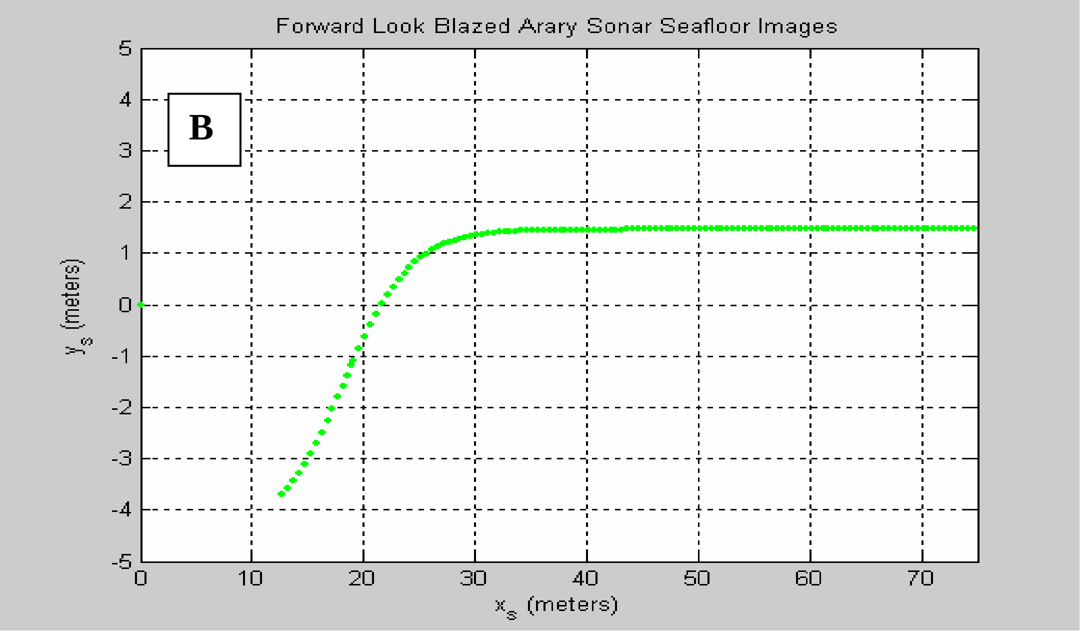


Figure 19. REMUS Dynamic Response Over Gradually Rising Sea Bottom

As mentioned earlier, forward look sonar undeniably enhances REMUS' obstacle avoidance capabilities. The following figures display chronological forward look sonar images of REMUS' ascent over a gradual 6 meter sea floor rise. These exact images are input and processed through REMUS' feedforward preview controller.


Forward Look Blazed Arary Sonar Seafloor Images
A
5
4
3
2
1
0
-1
-2
-3
-4
-5
0
10
20
30
40
50
60
70
y_s (meters)
x_s (meters)


Forward Look Blazed Arary Sonar Seafloor Images
B
5
4
3
2
1
0
-1
-2
-3
-4
-5
0
10
20
30
40
50
60
70
y_s (meters)
x_s (meters)

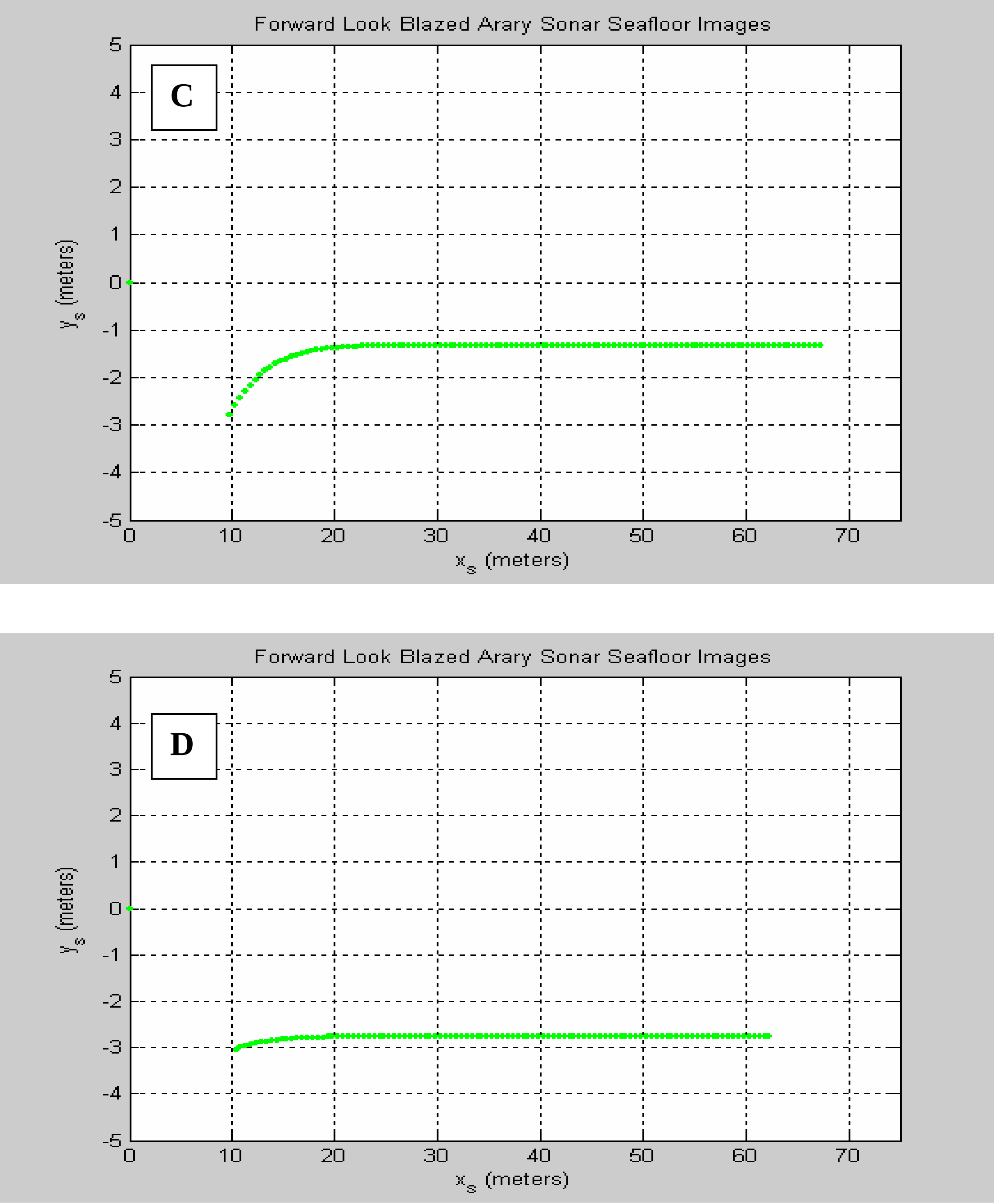


Figure 20. Sequential Display of Forward Look Sonar Images of Gradual 6 meter Sea Floor Rise

Notice that in image A in Figure 20, REMUS is approximately 60 meters from the 6 meter obstacle. At this point, REMUS' feedforward preview controller begins to provide the vehicle with adjusted trajectory commands causing REMUS to pitch upwards.. Image B and C display two different elevations during REMUS climb over the obstacle. Finally, Image D displays REMUS regaining its commanded 3 meter altitude after successfully navigating over the obstacle.

3. REMUS' Dynamic Response Over a 6 Meter Sharp Step

The following figure illustrates REMUS' dynamic response over a sharp 6 meter sea floor rise.

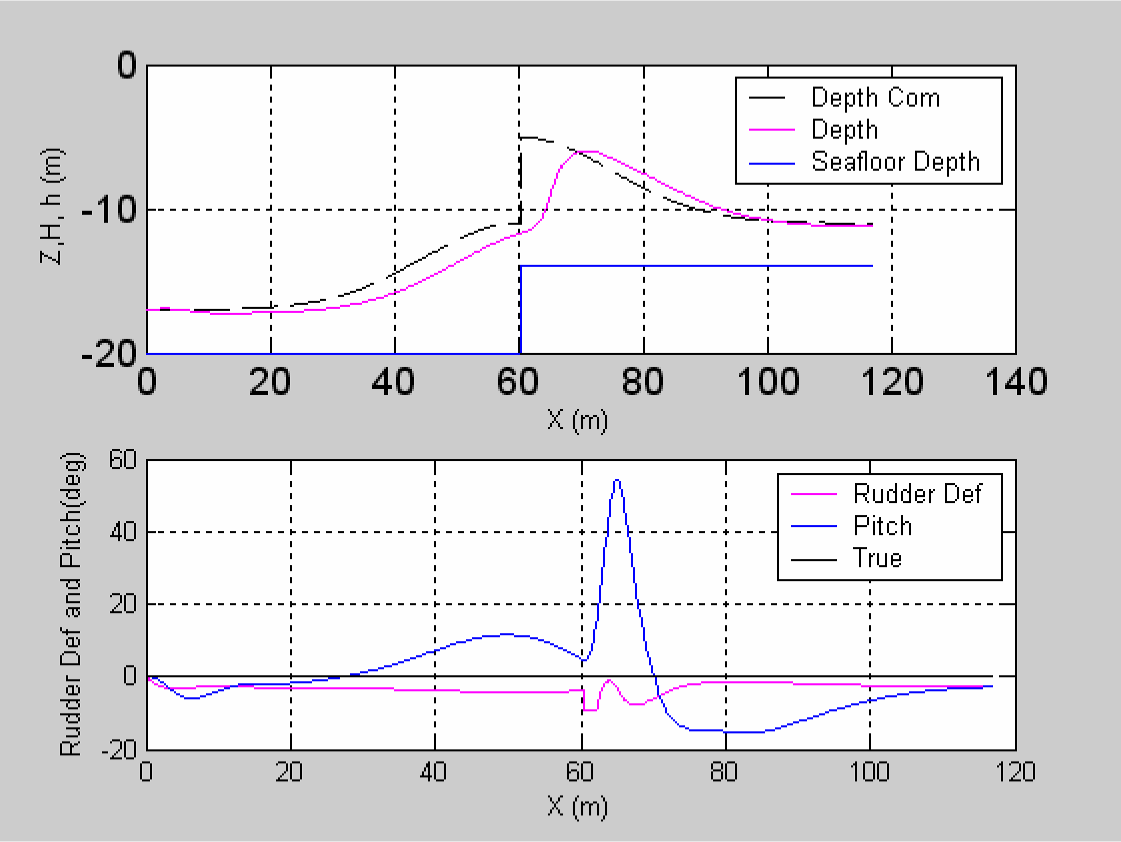
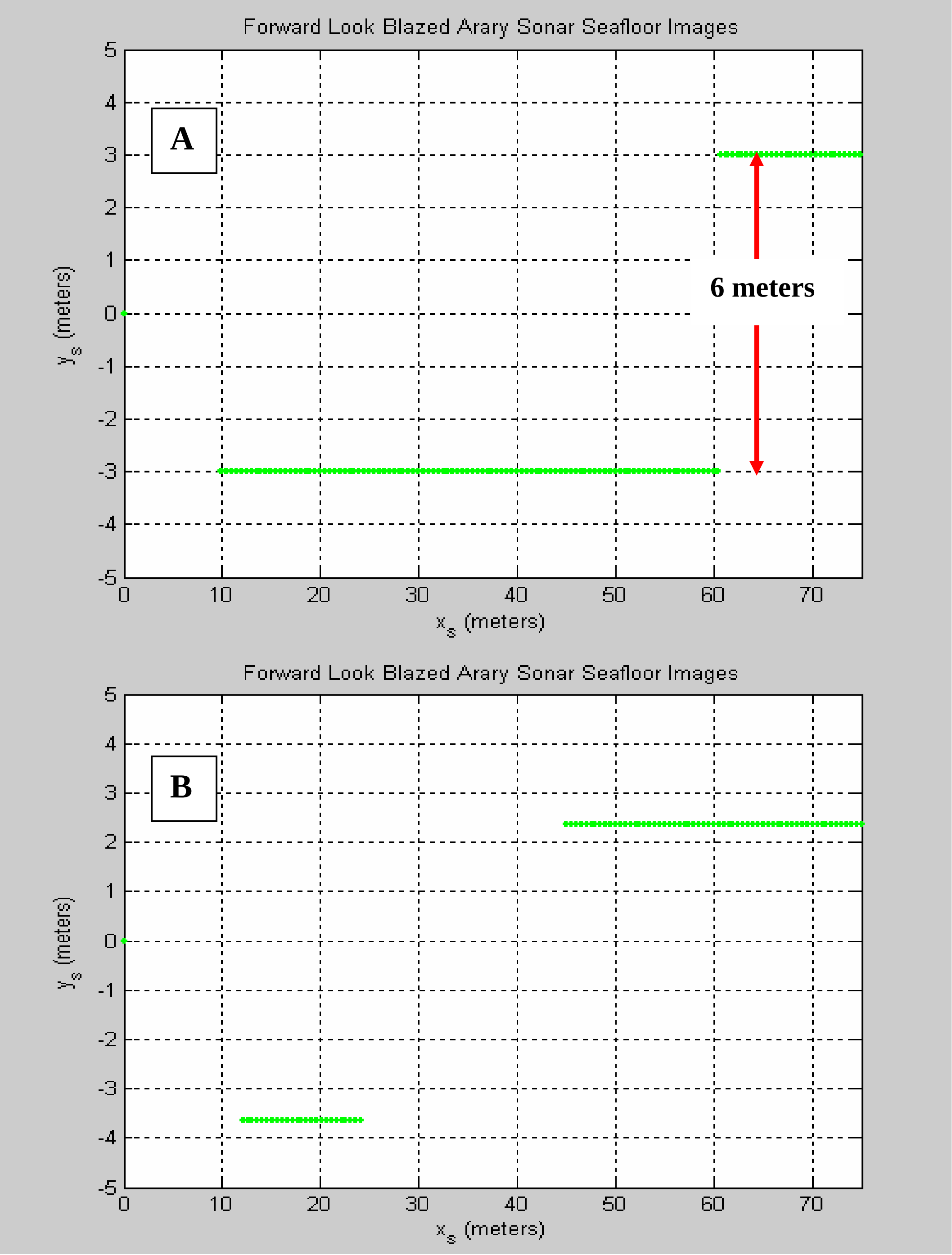


Figure 21. REMUS Dynamic Response over Sharp Step

Notice that in the above figure, REMUS begins climbing over the obstacle at roughly 35 meters from the global origin. Even though a small lag occurs between the obstacle avoidance controller's depth commands and the vehicle's actual response, REMUS manages to successfully avoid a collision. During this simulation, REMUS gathered the following forward look sonar images.


Forward Look Blazed Arary Sonar Seafloor Images
A
6 meters
y_s (meters)
x_s (meters)
5
4
3
2
1
0
-1
-2
-3
-4
-5
0
10
20
30
40
50
60
70
Forward Look Blazed Arary Sonar Seafloor Images
B
y_s (meters)
x_s (meters)

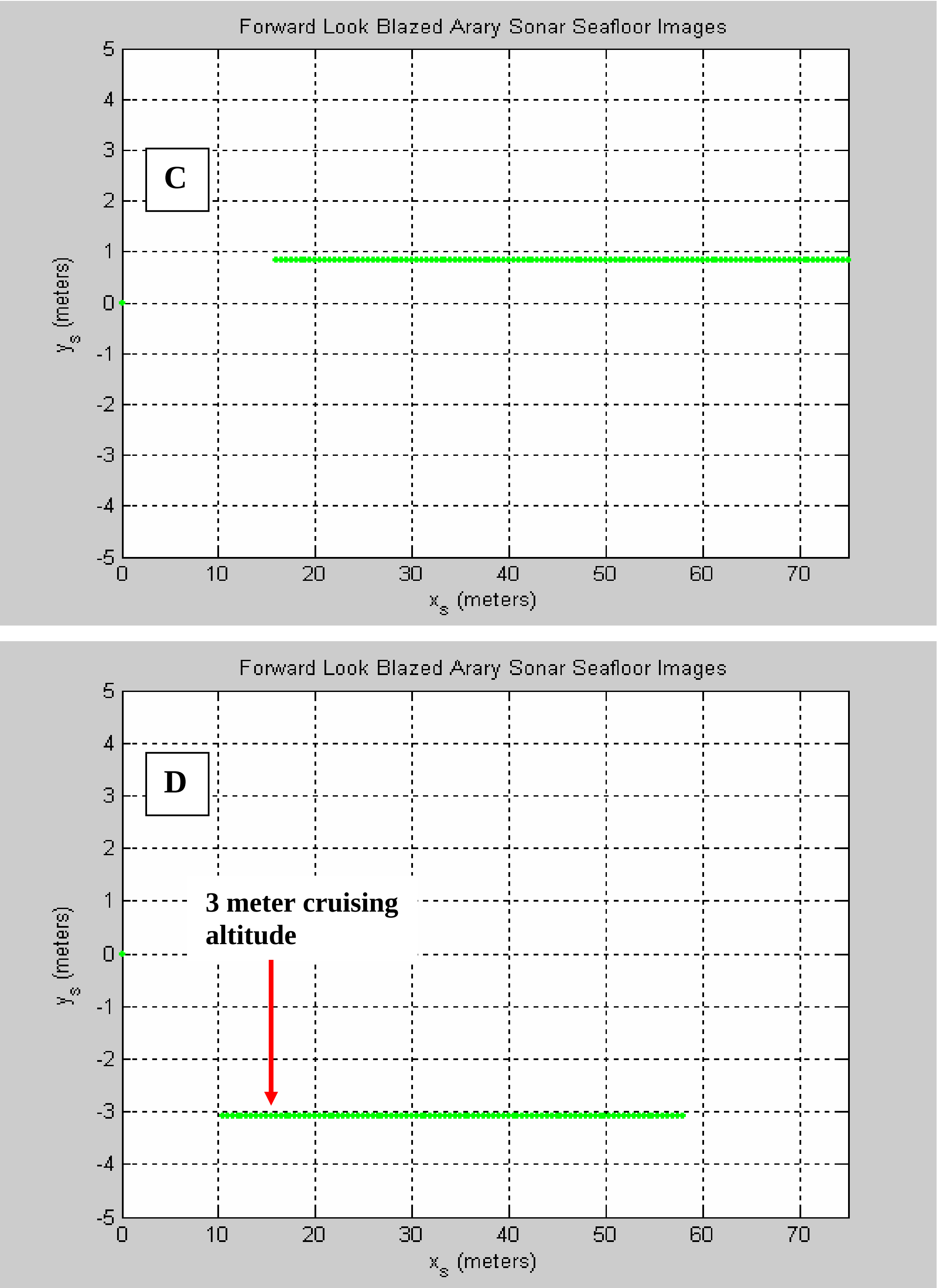


Figure 22. Sequential Display of Forward Look Sonar Images over a 6 meter High Sharp Step

Notice that image A accurately displays the height and range of the oncoming obstacle. As a result, REMUS' feedforward preview controller forces the vehicles stern control surfaces to deflect effectively pitching the vehicle upwards. Tracing the outline of the Gaussian potential field, REMUS continues to climb as shown in images B and C until finally leveling out. The final image confirms that REMUS has returned to its commanded 3 meter cruising altitude.

Examining the exact same simulation without the benefit of feedforward preview sonar image processing produces the following results.

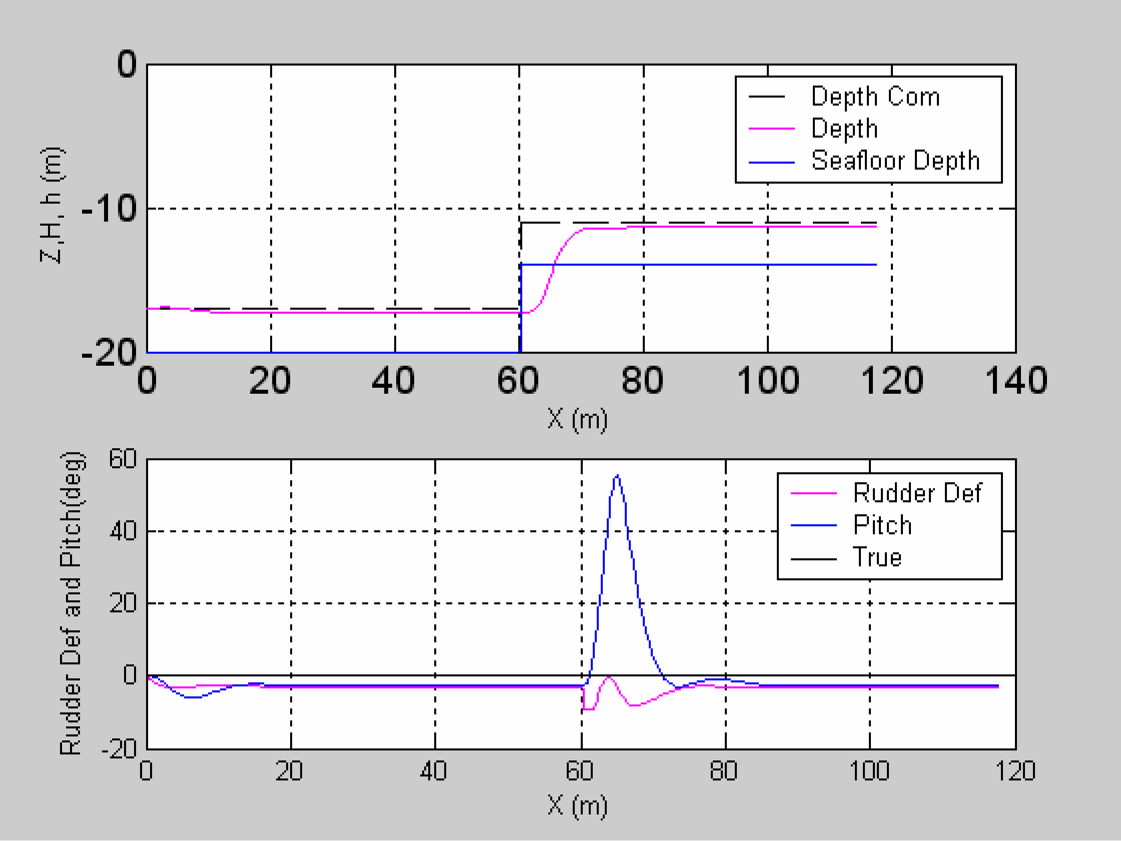


Figure 23. Dynamic Response of REMUS over 6 meter Sharp Step; No Forward Look Sonar

Notice that without processed forward look sonar images, REMUS is given no time to react to the abrupt change in sea floor elevation and therefore crashes.

4. REMUS' Dynamic Response Over a 6 Meter Sea Floor Hill

Now that REMUS has encountered sea bottoms that rise and level out at new, higher elevations, the following simulations involve sea bottoms that rise and then return to their original elevation. Unlike before, REMUS' obstacle avoidance controller will have to contend with slopes on either side of the obstacle. Figure 24 displays REMUS' adjusted trajectory over a simulated 6 meter high sea floor hill.

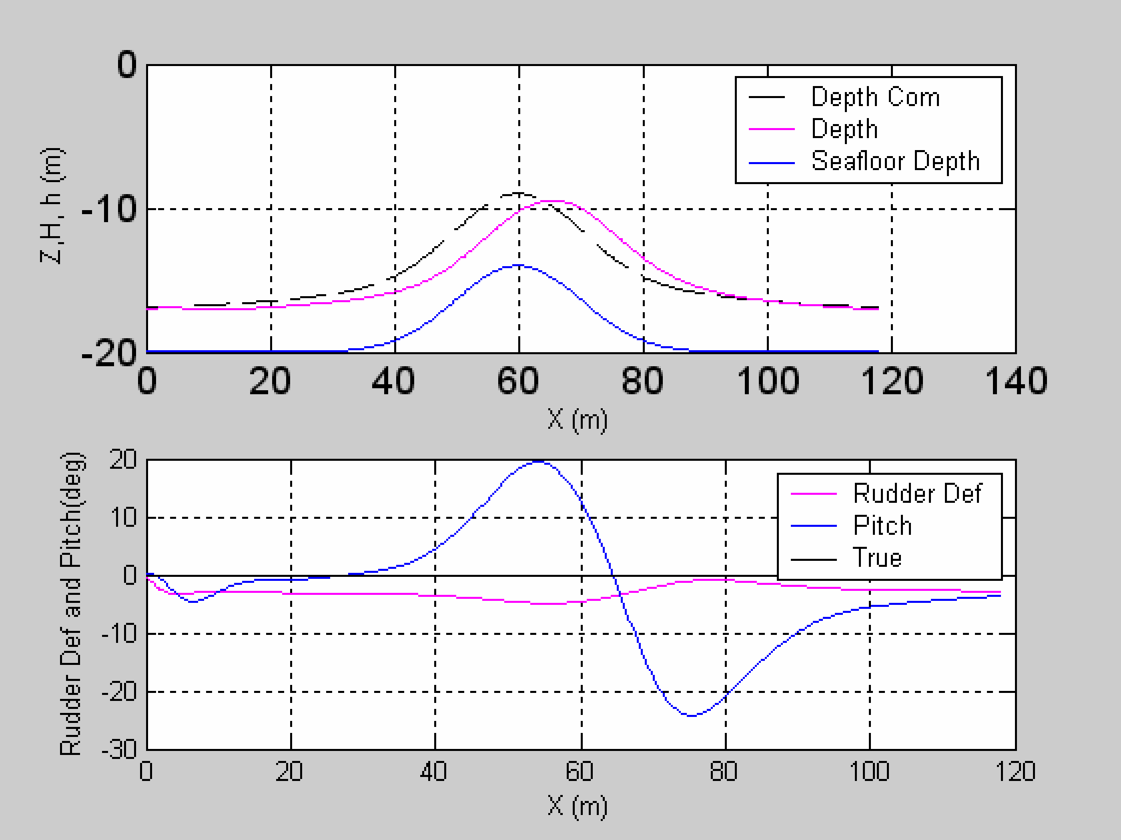
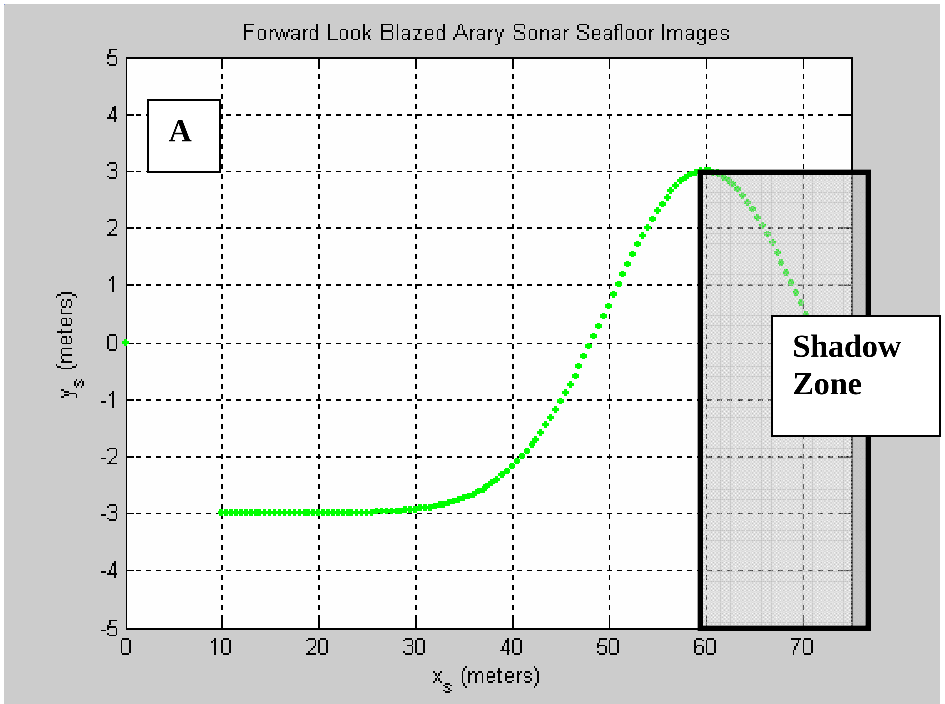
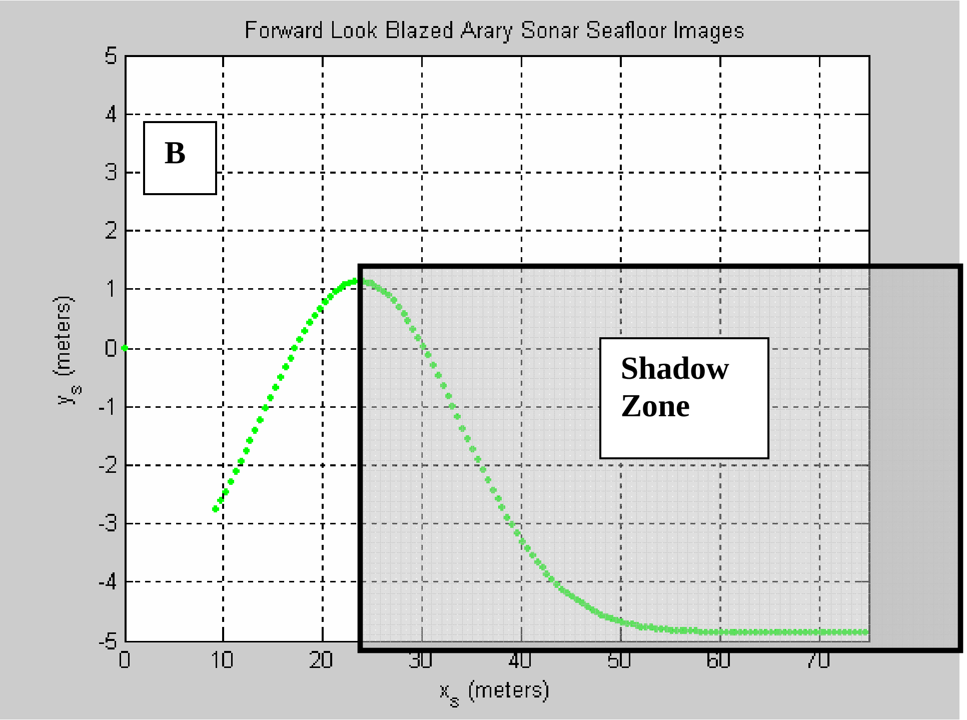


Figure 24. Dynamic Response of REMUS over 6 meter Sea Floor Hill

As illustrated in the above figure, REMUS' trajectory is smooth as the vehicle successfully navigates over then down the 6 meter hill. As REMUS was approaching and navigating over the hill, its forward look sonar gathered the following images.


Forward Look Blazed Arary Sonar Seafloor Images
A
Shadow Zone
y_s (meters)
x_s (meters)
5
4
3
2
1
0
-1
-2
-3
-4
-5
0
10
20
30
40
50
60
70


Forward Look Blazed Arary Sonar Seafloor Images
B
Shadow Zone
y_s (meters)
x_s (meters)
5
4
3
2
1
0
-1
-2
-3
-4
-5
0
10
20
30
40
50
60
70

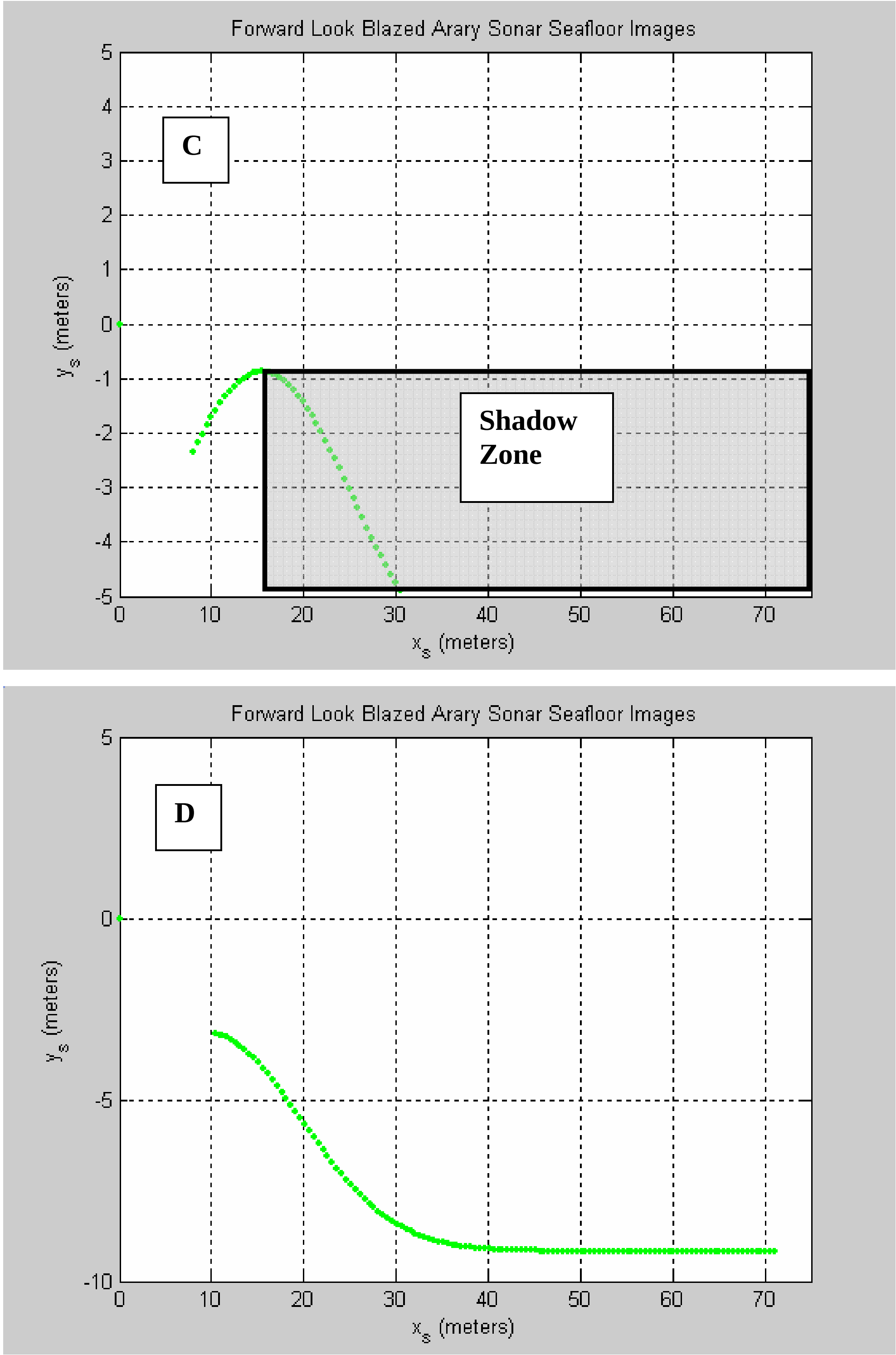


Figure 25. Sequential Display of Forward Look Sonar Images over a 6 meter Sea Floor Hill

Image A very clearly displays a 6 meter hill, however, MATLAB’s rendering of the simulated sea floor is not entirely correct. Realistically, forward look sonar is only able to gather range and bearing data for objects within its direct line of sight. In other words, if an object lies behind another object, it is located in an area commonly referred to as a shadow zone and is in essence invisible. Therefore, images A, B, and C are not entirely accurate because each image displays the backside of the approaching hill, whereas during actual missions, this would not be the case. Nevertheless, this inaccuracy does not affect the validity of this simulation since each image clearly provides the obstacle’s range and height with respect to REMUS’ real time position. With these two key pieces of information, REMUS’ feedforward preview controller is able to provide a suitable avoidance trajectory. Notice that image D does not contain a shadow zone. Since REMUS is above the obstacle, the backside slope of the hill is in the forward look sonar’s direct line of sight and is therefore visible.

5. REMUS’ Dynamic Response Over 6 Meter Sea Wall

The final simulation examines REMUS’ ability to navigate over a simulated 6 meter high sea wall. Similar to the sharp step sea floor, a sea wall is perhaps one of the toughest obstacles to avoid simply because such an abrupt change in sea floor elevation severely limits a vehicle’s reaction time. The following figure illustrates REMUS’ trajectory over a sea wall while relying on processed forward look sonar images.

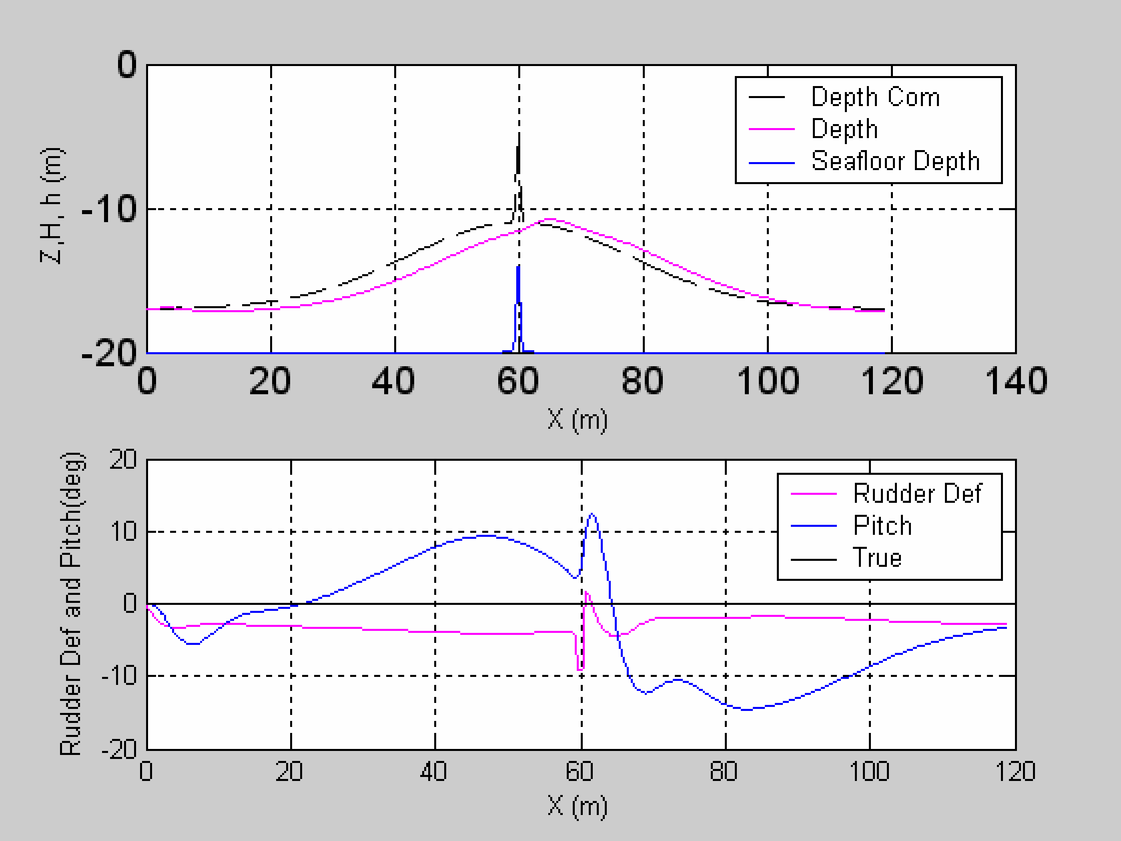


Figure 26. Dynamic Response of REMUS over 6 meter Sea Wall

Figure 26 provides possibly the best example of REMUS' feedforward preview controller obstacle avoidance capabilities. REMUS clearly traces a Gaussian potential field created by its onboard feedforward preview controller while it navigates safely over the 6 meter high obstacle. Vehicle trajectory was kept relatively smooth and gradual given pitch commands of no greater than ± 10 degrees. Contrastingly, Figure 28 displays REMUS' trajectory over the exact same sea wall without the benefit of feedforward preview control.

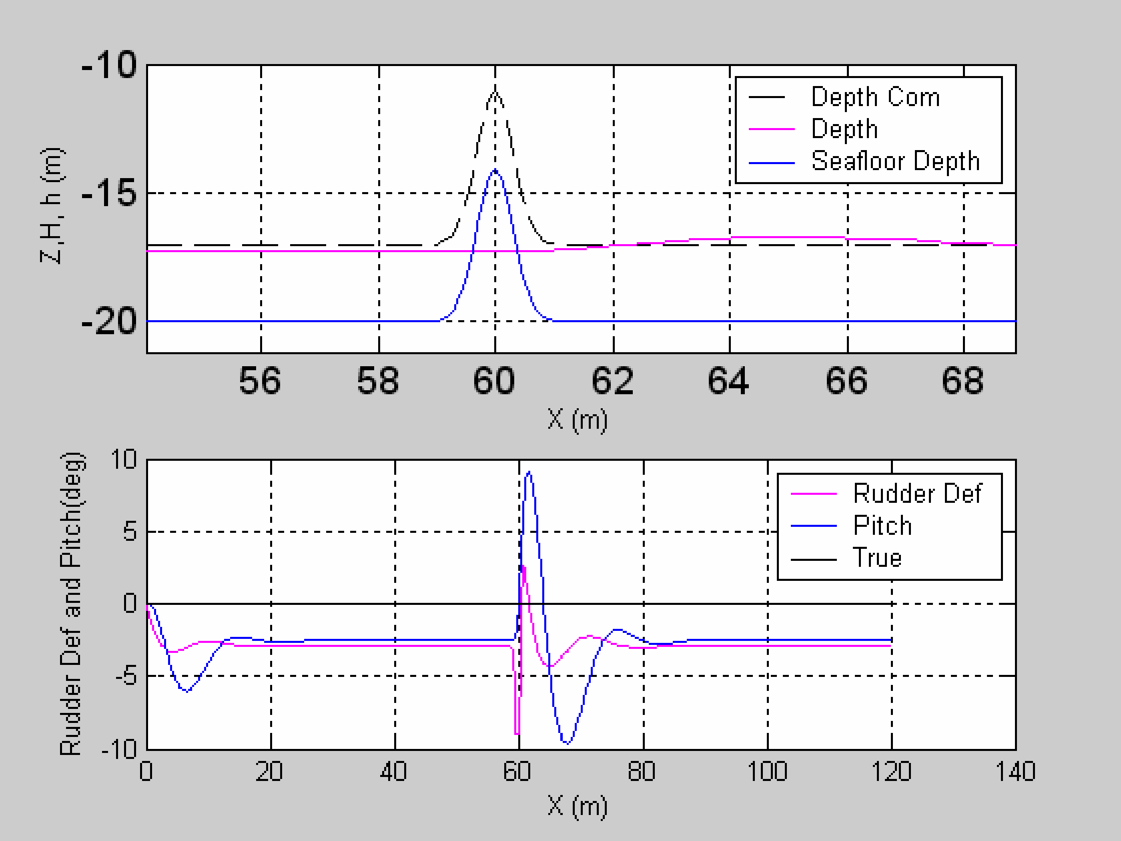
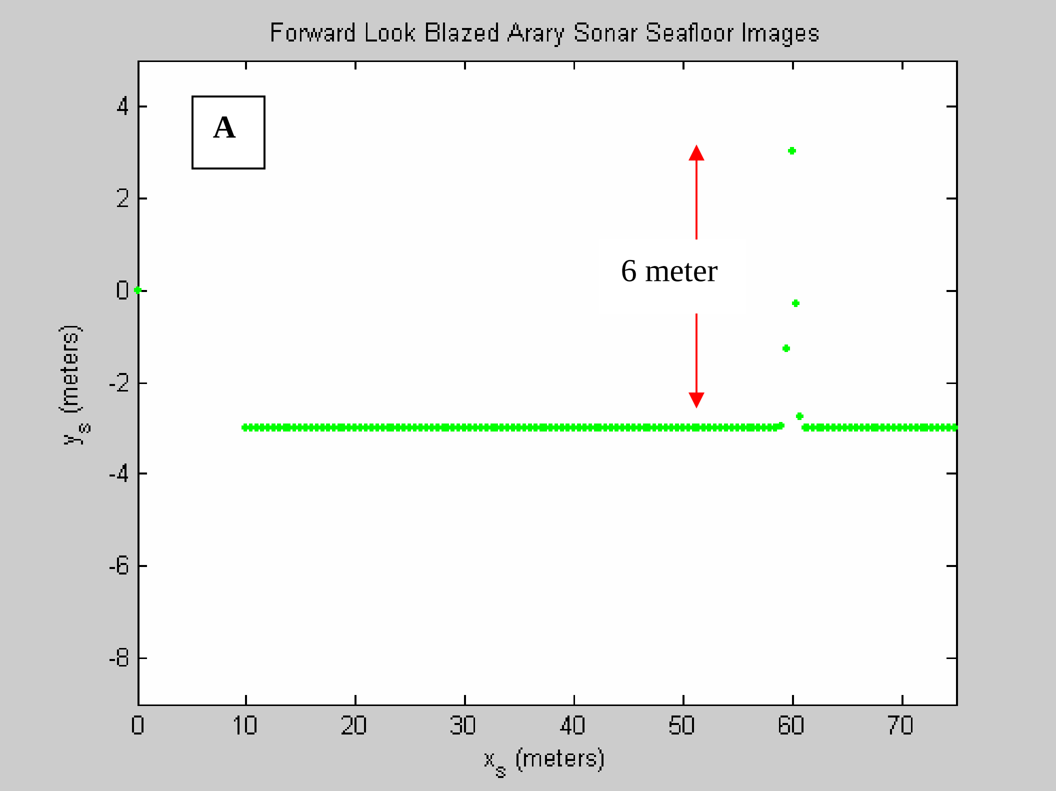
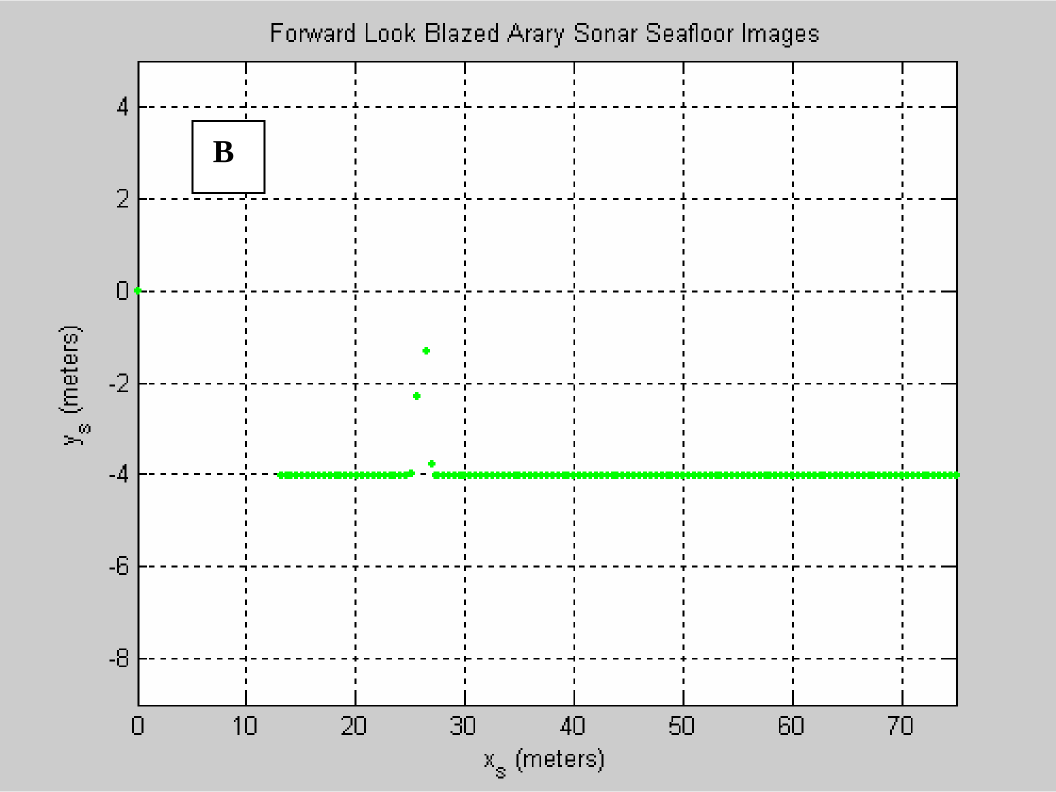


Figure 27. Dynamic Response of REMUS over 6 meter Sea Wall with Forward Look Sonar Turned off

In order to clearly illustrate the collision that occurs between REMUS and the sea wall, the top graph as been magnified. Without the benefit of processed forward look sonar images, rudder deflection commands arrive too late and therefore have no significant affect on REMUS' trajectory. Although the commanded depth supplied by the sliding mode autopilot does in fact avoid the sea wall, REMUS can not realistically follow this trajectory. Inherent lags regarding REMUS' response to control plane deflections coupled with certain maneuverability and environmental constraints resulting from inertia and added mass require a more robust and capable obstacle avoidance controller. Simply relying upon RDI Doppler altitude data is insufficient.


Forward Look Blazed Arary Sonar Seafloor Images
A
6 meter
y_s (meters)
x_s (meters)
4
2
0
-2
-4
-6
-8
0
10
20
30
40
50
60
70


Forward Look Blazed Arary Sonar Seafloor Images
B
y_s (meters)
x_s (meters)
4
2
0
-2
-4
-6
-8
0
10
20
30
40
50
60
70

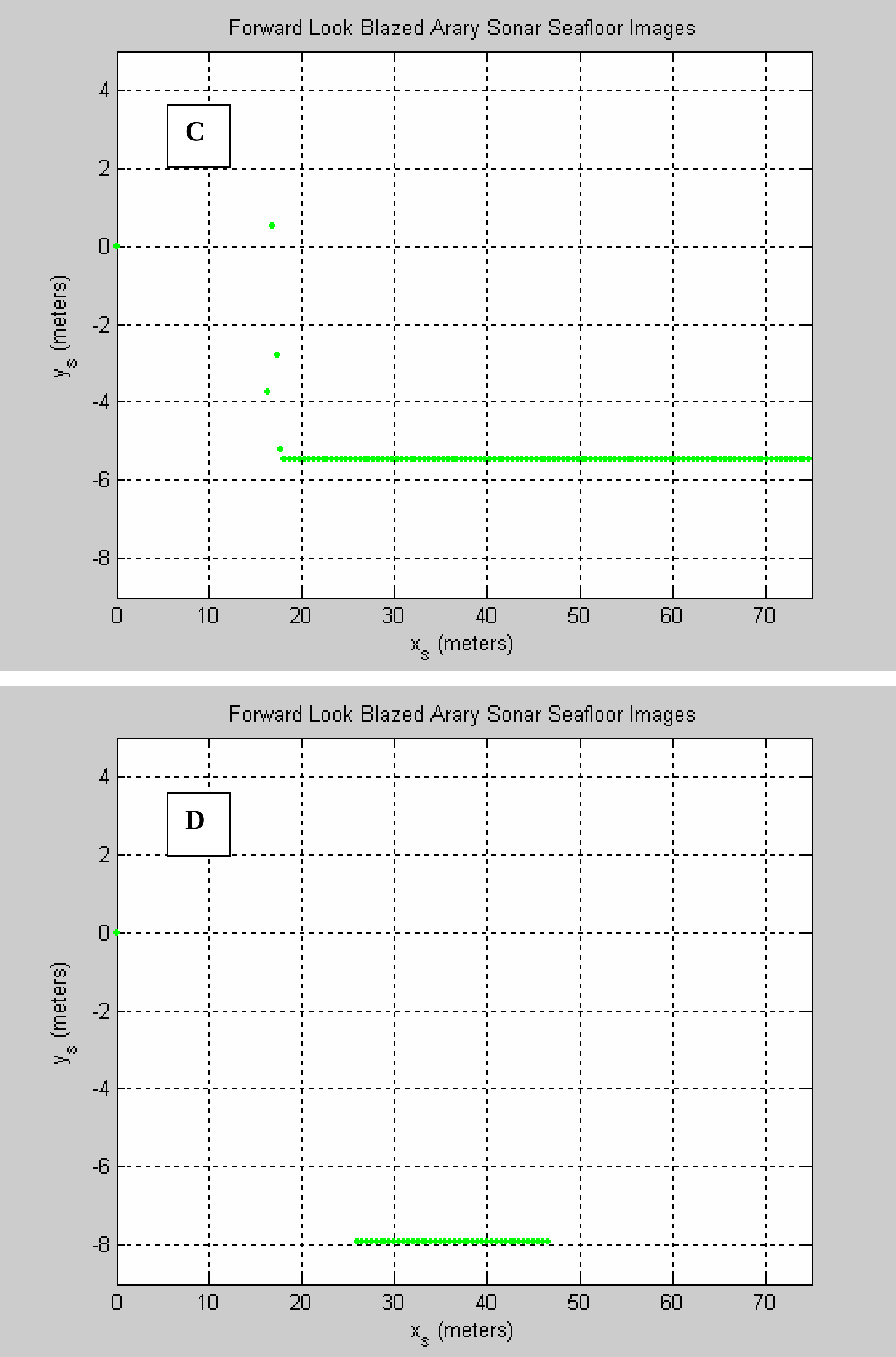


Figure 28. Sequential Display of Forward Look Sonar Images over 6 meter Sea Wall

Figure 28 depicts 4 sonar images collected by REMUS' forward look sonar. The grid in image A was removed allowing easier identification of the sea wall's initial height.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSION AND RECOMMENDATIONS

A. CONCLUSION

The work of this thesis involving vertical plane obstacle avoidance simulation using REMUS proves that equipping all AUVs with forward look sonar is essential. Forward look sensing capability allowed REMUS to avoid four different 6 meter obstacles that earlier were unavoidable using only RDI Doppler input. As it turns out, the optimum obstacle avoidance control algorithm involved collecting vertical plane forward look sonar images of simulated sea bottoms and processing them through a feedforward preview controller. This controller then assigned certain values to parameters within a Gaussian potential function according to the defining characteristics of the approaching obstacle. Summed with an altitude command of 3 meters, the Gaussian potential field defined an appropriate obstacle avoidance trajectory over each simulated obstacle. As discussed earlier, initial reliance upon half of a Gaussian potential field was abandoned in favor of using full Gaussian potential fields simply because the latter method produced more efficient trajectories over a wider range of sea floors.

In real world environments, having the ability to autonomously navigate within unfamiliar surroundings is critical to the success of any mission. The results of this thesis confirm that through forward look sonar image processing, we can in theory equip an AUV with enough intelligence to successfully maneuver in the vertical plane to avoid obstacles along that vehicles track.

B. RECOMMENDATIONS

Much more work is needed to fully solve the autonomous obstacle avoidance problem. Although this thesis theoretically proves that AUVs can in fact avoid obstacles in the vertical plane, further testing needs to travel beyond this solution. Simulation can model real ocean environments to a certain degree; however they simply cannot account for every subtlety an AUV could possibly encounter during a mission. Therefore, in order to truly test the concepts presented in this thesis, analysis involving the theoretical obstacle avoidance model defined in this thesis along with real forward look sonar data should be completed. Secondly, avoidance controllers can always be improved. More robust controllers compatible with a wider range of sea floors and ocean obstacles are needed.

In the future, more research and testing should also involve equipping AUVs with more than one forward look sonar. Individually configuring each sonar along different sections of the AUV could in essence provide a full 360 degree 3-D field of view. As a result, blind spots could be eliminated altogether and AUVs could in theory avoid any obstacle within its operational environment regardless of that obstacles orientation relative to the vehicle.

APPENDIX I: MATLAB CODES FOR GRADUAL RISE/STEP USING FULL POTENTIAL FUNCTION

remusderivalt.m

```
function[xdot,ds,sig,sigdot,h,TRUE,depthcom,range,bearing,threat] =
remusderivalt(t,xx);
%
%
% remusderivalt is an smc controller that is called up by an
% ode function commanding the vehicle to a specific altitude.
% Created by Chris Chuhran, May 1, 2003

% REMUS parameters ---------------------------------------------------

U = xx(6);
s = xx(7:10);
k = xx(11:14)';
x = xx(1:4);        % x(1) = q, x(2) = w, x(3) = theta, x(4) = Z, xx(5)
= X
xcom = xx(15:18);   % xcom = [q com, w com, theta com, depth com]
z_g = 1.96e-2;
x_b = 0;
W = 299;
buoy = 306;

global TRUE;
global DDIST;
global HEIGHT;

I_z = 3.45;
I_y = 3.45;
I_x = 1.77e-1;
U = 1.5;
m = 299/9.81;
M_q = -6.87;
M_qdot = -4.88;
M_w = 30.7;
M_wdot = -1.93;
M_d = -34.6;
Z_q = -9.67;
Z_qdot = -1.93;
Z_w = -66.6;
Z_wdot = -35.5;
Z_d = -50.6;


thetacom = 0;
altcom = 3;

Ra = 20;            % Sonar Range (m)
```

```
%SSTART = 60 - Ra;   % this variable needs to be named once for each
obstacle, hardwired for now
% Dynamics -------------------------------------------------------------
-
M = [m-Z_wdot -Z_qdot 0 0;-M_wdot I_y-M_qdot 0 0;0 0 1 0;0 0 0 1];
A_0 = [Z_w m*U+Z_q 0 0 ;M_w M_q -z_g*W 0;0 1 0 0;1 0 -U 0];
B_0 = [Z_d;M_d;0;0];

A = inv(M)*A_0;
B = inv(M)*B_0;
C = [0 0 0 1];
D = inv(M)*[-7;0;0;0];

% Seafloor Modeling for Sonar (non-time dependent) ---------------------
---
% Seabottom I ----------------------------------------------------------
---
% X_1 = [0:0.5:60];
% X_2 = [60:0.2:62.8];
% X_3 = [62.8:0.5:65.8];
% X_4 = [65.8:0.2:68.6];
% X_5 = [68.6:0.5:111.6];
% X_Model = [X_1 X_2 X_3 X_4 X_5];
%
% H_1 = 20*ones(1,121);
% H_2 = [20:-0.5:13];
% H_3 = 13*ones(1,7);
% H_4 = [13:0.5:20];
% H_5 = 20*ones(1,87);
% H_Model = [H_1 H_2 H_3 H_4 H_5];

% Seabottom II ---------------------------------------------------------
---



X_1 = [0:0.5:60];
X_2 = [60:0.4:61.4];
X_3 = [61.4:0.5:121.4];
X_Model = [X_1 X_2 X_3];
S=6;la=.01;
H_Model=seabottom(X_Model,S,la);

bearing=zeros(1,length(X_Model));
range=bearing;

TRUE = 0;
% Sonar ----------------------------------------------------------------
--
for d = 1:length(X_Model)
    if X_Model(d) > xx(5)
       r = sqrt((X_Model(d) - xx(5))^2 + (H_Model(d)-x(4))^2);
        b = asin((H_Model(d) - x(4))/r) ;      % bearing to object as
read by sonar (pitch corrected)
    %
```

```
%         floor_brng = 3*pi/180;         % this is bearing of LF beam
at 3 m alt
%
%         if (b) == 0         % prevents divide by zero error
(sin(angle))
%             floor_alt = 100;             % this happens when obstacle
is directly in front of REMUS, b=0the 100 is arbitrary
%         else
%             floor_alt = 3/sin(b );     % this is range to ocean floor
minus buffer
%         end                                          % buffer of 0.6
can be handled by altitude control
        bearing(d) = 0;range(d) = 0; % initialization
        if (b > -3*pi/180 & b < 17*pi/180 & r < 100)

            %TRUE = 1;  % this means the bottom grid point lies in the
active zone
            bearing(d) = b; range(d) = r; % this registers the range
and bearing of the bottom data point
            %DDIST = r + xx(5) + 0;    % ensures no dive before
obstacle is passed not used

%             for dd = d:length(X_Model)  old stuff from Chris work
%                 if abs(H_Model(dd) - H_Model(dd-1)) <= 0.001
%                     HEIGHT = 20 - H_Model(dd);
%                     break
%                 end
%             end

            %break

        else TRUE = 0;
        end
    end
end


% - Threat Level Assessment


wr = zeros(1,length(range));
wb = wr;
wcount=0;
bcount=0;
for i=1:length(range),
        wr(i) = rangeweight(range(i));if (abs(wr(i))>0),
wcount=wcount+1;end;
        wb(i) = bweight(bearing(i));if
(abs(wb(i))>0),bcount=bcount+1;end;
end;
threat=0;
%if (wcount>0 & bcount>0), threat  =
(wr*wr')/wcount*(wb*wb')/bcount;end;
```

```
%  if (xx(3)< -5*pi/180), threat = 0;end; % put in to help on bouncing
```

```
% Controller -------------------------------------------------------
--
% if ((TRUE == 1)| (xx(5) < DDIST))
%
%     %altcom = 3 + (xx(5) - SSTART)*HEIGHT/(35-12);%ramping altitude
command
sig=15;
altcom=3+6*exp(-(xx(5)-60.0)^2/2/sig^2);
    %elseif ((xx(5)-60.0)) > 0 altcom=3;
    %6.0*threat;
% %     if altcom > HEIGHT + 3
% %        altcom = 3 + HEIGHT;
% %     elseif altcom < 3
% %        altcom = 3;
% %     end
% end;

% Seafloor Modeling for Controller (time dependent)
% Seabottom I ------------------------------------------------------
---
% if xx(5) <= 60 | xx(5) >= 68.6
% H = 20;
% elseif (xx(5) > 60 & xx(5) <= 62.8)
%     H = 170 - 2.5*xx(5);
% elseif xx(5) > 62.8 & xx(5) <= 65.8
%     H = 13;
% elseif xx(5) >65.8 & xx(5) < 68.6
%     H = -151.5 + 2.5*xx(5);
% end

% Seabottom II -----------------------------------------------------
----
% if xx(5) <= 60
%     H = 20;
% elseif (xx(5) > 60 & xx(5) <=61.4)
%     %elseif (xx(5) > 60 & xx(5) <=62)
%      %H = 140 - 2*xx(5);        % depth = 16m
%     H = 320 - 5*xx(5);        % depth = 13m
% elseif xx(5) > 61.4
%     H = 14;
% end
H=seabottom(xx(5),S,la);
depthcom = H - altcom;            % altitude control must be converted
to depth control for EOM

% % if ((TRUE == 1) | (xx(5) < DDIST))          % prevents jump up at
edge
```

```
% %     depthcom = 20 - altcom;                  % hardwired for now, 
need to "look back"
% % end

xcom=[0;0;thetacom;depthcom];
 phi = 0.1;
 sig=s'*(x-xcom);
 Nmax= 2;
 %ada = Nmax*0.4/inv((s'*B));

%delta = -k*x-Nmax*0.4*sign(inv((s'*B)))*tanh((sig/phi));
%..........smc controller

xe=(x-xcom);
delta = -k*xe;    %............LQR Controller


if abs(delta) > 0.157                          % REMUS has nine deg max 
rudder deflection
    delta = 0.157*sign(delta);
end

h = H - x(4);                                  % depth for plotting 
purposes
ds = delta;                                    % rudder angle for plotting 
purposes
xsdot = A*x+B*ds+D;
sigdot = s'*xsdot;
xsdot(4) = [x(1)*cos(x(3))-U*sin(x(3))];   % Large angle approximation
xxdot = [U*cos(x(3))+x(1)*sin(x(3))];      % Horizontal advance
xdot = [xsdot;xxdot;0;0;0;0;0;0;0;0;0;0;0;0;0];
```

REMUSCHRIS.m

```
clear
clc
z_g = 1.96e-2;
x_b = 0;
W = 299;
buoy = 306;
I_z = 3.45;
I_y = 3.45;
I_x = 1.77e-1;
U = 1.5;
to = 0;
tf = 80;

global TRUE;
global DDIST;
global HEIGHT;

TRUE = 0;
DDIST = 0;
HEIGHT = 0;
m = 299/9.81;
M_q = -6.87;
M_qdot = -4.88;
M_w = 30.7;
M_wdot = -1.93;
M_d = -34.6;
Z_q = -9.67;
Z_qdot = -1.93;
Z_w = -66.6;
Z_wdot = -35.5;
Z_d = -50.6;

% Dynamics ------------------------------------------------------------
M = [m-Z_wdot -Z_qdot 0 0;-M_wdot I_y-M_qdot 0 0;0 0 1 0;0 0 0 1];
A_0 = [Z_w m*U+Z_q 0 0;M_w M_q -z_g*W 0;0 1 0 0;1 0 -U 0];
B_0 = [Z_d;M_d;0;0];

A = inv(M)*A_0;
B = inv(M)*B_0;
C = [0 0 0 1];
D = inv(M)*[-7;0;0;0];
Q1=diag([1,1,4,1]);R1=400;

% Pole Placement ------------------------------------------------------
-
p = [0 -0.6 -0.62 -0.63];
k = place(A,B,p);
klqr = lqr(A,B,Q1,R1)
Ac = A-B*k;
eig(A-B*klqr)
[V,v] = eig(Ac');
s = V(:,4);

% Controller ---smc----------------------------------------------------
------
x0 = [0;0;0;17;0;U;s;k';[0;0;0;3]];                       % initial
condition and command
% Controller lqr
x0 = [0;0;0;17;0;U;0;0;0;0;klqr';[0;0;0;3]];
[t,x] = ode45(@remusderivalt,[to tf],x0);

TRUE = 0;
DDIST = 0;
HEIGHT = 0;
R=zeros(length(t),100);
B=R;th=zeros(1,length(t));
for i = 1:length(t)
```

```
[xdot,ds,sig,sigdot,h,TRUE,depthcom,range,bearing,threat]=remusderivalt
(t(i),x(i,:)');
    T(i) = TRUE;
    DEP(i) = -depthcom;
    range;
    bearing;
    th(i) = threat;
        for j=1:length(range)
        R(i,j) = range(j);
        B(i,j) = bearing(j);
        end;
    sigma(i) = sig;
    alt(i) = h;
    deltasp(i) = ds*180/pi;
    H(i) = alt(i) + x(i,4);
end;

% Plotting ------------------------------------------------------------
---------
%convert to xy coordinates from rb space
ys=R(:,:).*sin(B(:,:));xs=R.*cos(B);
%plot
  figure(1),clf;
     subplot(2,1,1),plot(x(:,5),DEP,'k--',x(:,5),-x(:,4),'m',x(:,5),-
H,'b'),grid

subplot(2,1,2),plot(x(:,5),deltasp,'m',x(:,5),x(:,3).*180/pi,x(:,5),T*2
0,'k'),grid


     subplot(2,1,1),xlabel('X (m)')
     subplot(2,1,1),ylabel(' Z,H, h (m)')
     subplot(2,1,1),legend('Depth Com','Depth','Seafloor Depth')
     subplot(2,1,2),xlabel('X (m)')
     subplot(2,1,2),ylabel('Rudder Def and Pitch(deg)')
     subplot(2,1,2),legend('Rudder Def','Pitch','True')
     subplot(2,1,1),axis([0 140 -20 0]),HT=gca;set(HT,'FontSize',14)

%figure(2),clf,plot(ys(100,:),xs(100,:),'b*'),grid,hold,axis('equal');

     for i=(1:length(B)-140);
          if (mod(i,10)==0)
               figure(2),clf,plot(xs(i,:),-ys(i,:),'g.'),axis([0,75,-
5,5]),grid,title('Forward Look Blazed Arary Sonar Seafloor
Images'),xlabel('x_s (meters)'),ylabel('y_s (meters)');
           pause;
          end
      end


figure(3),clf,plot(x(:,5),th,'r.'),grid;
title('Threat Level as a Function of Distance Along Track');
```

seabottom.m

```
function H=seabottom(X,S,l);
if (X<= 60), H=20;end;

    if (X>= 60+l), H=20-S;end;
        if (60<X<60+l), H=20-S*.5*(1+tanh((X-60-.5*1)/l));
        end;
```

bweight.m

```
function wb = bweight(b)

b=abs(b)*180/pi;

    if b < 10, wb = 1;end;
    if b > 20,wb = 0;end;
    if b >= 10 & b < 20, wb = 1-(b-10)/10;end;
```

rangeweight.m

```
function w=rangeweight(r)

    if r<0, w=0;end;
    if r == 0, w = 0;end
    if r < 5 & r>0, w = 1;end;
    if r >20,w = 0;end;
    if r >= 5 & r < 20, w = 1-(r-5)/15;end;
```

APPENDIX II: MATLAB CODES FOR GRADUAL RISE/STEP USING HALF POTENTIAL FUNCTION

remusderivalt_halfpotential.m

```
function[xdot,ds,sig,sigdot,h,TRUE,depthcom,range,bearing,threat] =
remusderivalt(t,xx);
%
%
% remusderivalt is an smc controller that is called up by an
% ode function commanding the vehicle to a specific altitude.
% Created by Chris Chuhran, May 1, 2003

% REMUS parameters ---------------------------------------------------

U = xx(6);
s = xx(7:10);
k = xx(11:14)';
x = xx(1:4);        % x(1) = q, x(2) = w, x(3) = theta, x(4) = Z, xx(5)
= X
xcom = xx(15:18);   % xcom = [q com, w com, theta com, depth com]
z_g = 1.96e-2;
x_b = 0;
W = 299;
buoy = 306;

global TRUE;
global DDIST;
global HEIGHT;

I_z = 3.45;
I_y = 3.45;
I_x = 1.77e-1;
U = 1.5;
m = 299/9.81;
M_q = -6.87;
M_qdot = -4.88;
M_w = 30.7;
M_wdot = -1.93;
M_d = -34.6;
Z_q = -9.67;
Z_qdot = -1.93;
Z_w = -66.6;
Z_wdot = -35.5;
Z_d = -50.6;


thetacom = 0;
altcom = 3;

Ra = 20;            % Sonar Range (m)
```

```
%SSTART = 60 - Ra;   % this variable needs to be named once for each
obstacle, hardwired for now
% Dynamics ---------------------------------------------------------------
-
M = [m-Z_wdot -Z_qdot 0 0;-M_wdot I_y-M_qdot 0 0;0 0 1 0;0 0 0 1];
A_0 = [Z_w m*U+Z_q 0 0 ;M_w M_q -z_g*W 0;0 1 0 0;1 0 -U 0];
B_0 = [Z_d;M_d;0;0];

A = inv(M)*A_0;
B = inv(M)*B_0;
C = [0 0 0 1];
D = inv(M)*[-7;0;0;0];

% Seafloor Modeling for Sonar (non-time dependent) --------------------
---
% Seabottom I ---------------------------------------------------------
---
% X_1 = [0:0.5:60];
% X_2 = [60:0.2:62.8];
% X_3 = [62.8:0.5:65.8];
% X_4 = [65.8:0.2:68.6];
% X_5 = [68.6:0.5:111.6];
% X_Model = [X_1 X_2 X_3 X_4 X_5];
%
% H_1 = 20*ones(1,121);
% H_2 = [20:-0.5:13];
% H_3 = 13*ones(1,7);
% H_4 = [13:0.5:20];
% H_5 = 20*ones(1,87);
% H_Model = [H_1 H_2 H_3 H_4 H_5];

% Seabottom II --------------------------------------------------------
---



X_1 = [0:0.5:60];
X_2 = [60:0.4:61.4];
X_3 = [61.4:0.5:121.4];
X_Model = [X_1 X_2 X_3];
S=6;la=6;
H_Model=seabottom(X_Model,S,la);

bearing=zeros(1,length(X_Model));
range=bearing;

TRUE = 0;
% Sonar ---------------------------------------------------------------
--
for d = 1:length(X_Model)
    if X_Model(d) > xx(5)
       r = sqrt((X_Model(d) - xx(5))^2 + (H_Model(d)-x(4))^2);
        b = asin((H_Model(d) - x(4))/r) ;      % bearing to object as
read by sonar (pitch corrected)
    %
```

```
%         floor_brng = 3*pi/180;         % this is bearing of LF beam
at 3 m alt
%
%         if (b) == 0         % prevents divide by zero error
(sin(angle))
%             floor_alt = 100;             % this happens when obstacle
is directly in front of REMUS, b=0the 100 is arbitrary
%         else
%             floor_alt = 3/sin(b );     % this is range to ocean floor
minus buffer
%         end                                          % buffer of 0.6
can be handled by altitude control
        bearing(d) = 0;range(d) = 0; % initialization
        if (b > -3*pi/180 & b < 17*pi/180 & r < 100)

            %TRUE = 1;  % this means the bottom grid point lies in the
active zone
            bearing(d) = b; range(d) = r; % this registers the range
and bearing of the bottom data point
            %DDIST = r + xx(5) + 0;    % ensures no dive before
obstacle is passed not used

%             for dd = d:length(X_Model)  old stuff from Chris work
%                 if abs(H_Model(dd) - H_Model(dd-1)) <= 0.001
%                     HEIGHT = 20 - H_Model(dd);
%                     break
%                 end
%             end

            %break

        else TRUE = 0;
        end
    end
end


% - Threat Level Assessment


wr = zeros(1,length(range));
wb = wr;
wcount=0;
bcount=0;
for i=1:length(range),
        wr(i) = rangeweight(range(i));if (abs(wr(i))>0),
wcount=wcount+1;end;
        wb(i) = bweight(bearing(i));if
(abs(wb(i))>0),bcount=bcount+1;end;
end;
threat=0;
if (wcount>0 & bcount>0), threat  =
(wr*wr')/wcount*(wb*wb')/bcount;end;
```

```
%  if (xx(3)< -5*pi/180), threat = 0;end; % put in to help on bouncing





% Controller ----------------------------------------------------------
--
% if ((TRUE == 1)| (xx(5) < DDIST))
%
%     %altcom = 3 + (xx(5) - SSTART)*HEIGHT/(35-12);%ramping altitude
command
sig=8;      %controls width of exponential function
if ((xx(5)-60.0)) < 0,    altcom=3+6*exp(-(xx(5)-60.0)^2/2/sig^2);
    elseif ((xx(5)-60.0)) > 0 altcom=3;
end;
%6.0*threat;
% %     if altcom > HEIGHT + 3
% %        altcom = 3 + HEIGHT;
% %     elseif altcom < 3
% %        altcom = 3;
% %     end
% end;

% Seafloor Modeling for Controller (time dependent)
% Seabottom I ---------------------------------------------------------
---
% if xx(5) <= 60 | xx(5) >= 68.6
% H = 20;
% elseif (xx(5) > 60 & xx(5) <= 62.8)
%     H = 170 - 2.5*xx(5);
% elseif xx(5) > 62.8 & xx(5) <= 65.8
%     H = 13;
% elseif xx(5) >65.8 & xx(5) < 68.6
%     H = -151.5 + 2.5*xx(5);
% end

% Seabottom II --------------------------------------------------------
----
% if xx(5) <= 60
%     H = 20;
% elseif (xx(5) > 60 & xx(5) <=61.4)
%     %elseif (xx(5) > 60 & xx(5) <=62)
%      %H = 140 - 2*xx(5);        % depth = 16m
%     H = 320 - 5*xx(5);        % depth = 13m
% elseif xx(5) > 61.4
%     H = 14;
% end
H=seabottom(xx(5),S,la);
depthcom = H - altcom;            % altitude control must be converted
to depth control for EOM

% % if ((TRUE == 1) | (xx(5) < DDIST))          % prevents jump up at
edge
```

```
% %     depthcom = 20 - altcom;                 % hardwired for now,
need to "look back"
% % end

xcom=[0;0;thetacom;depthcom];
 phi = 0.1;
 sig=s'*(x-xcom);
 Nmax= 2;
 %ada = Nmax*0.4/inv((s'*B));

%delta = -k*x-Nmax*0.4*sign(inv((s'*B)))*tanh((sig/phi));
%..........smc controller

xe=(x-xcom);
delta = -k*xe;    %............LQR Controller


if abs(delta) > 0.157                      % REMUS has nine deg max
rudder deflection
    delta = 0.157*sign(delta);
end

h = H - x(4);                              % depth for plotting
purposes
ds = delta;                                % rudder angle for plotting
purposes
xsdot = A*x+B*ds+D;
sigdot = s'*xsdot;
xsdot(4) = [x(1)*cos(x(3))-U*sin(x(3))];   % Large angle approximation
xxdot = [U*cos(x(3))+x(1)*sin(x(3))];      % Horizontal advance
xdot = [xsdot;xxdot;0;0;0;0;0;0;0;0;0;0;0;0;0];
```

REMUSCHRIS.m

```
clear
clc
z_g = 1.96e-2;
x_b = 0;
W = 299;
buoy = 306;
I_z = 3.45;
I_y = 3.45;
I_x = 1.77e-1;
U = 1.5;
to = 0;
tf = 80;

global TRUE;
global DDIST;
global HEIGHT;

TRUE = 0;
```

```
DDIST = 0;
HEIGHT = 0;
m = 299/9.81;
M_q = -6.87;
M_qdot = -4.88;
M_w = 30.7;
M_wdot = -1.93;
M_d = -34.6;
Z_q = -9.67;
Z_qdot = -1.93;
Z_w = -66.6;
Z_wdot = -35.5;
Z_d = -50.6;

% Dynamics -------------------------------------------------------------
M = [m-Z_wdot -Z_qdot 0 0;-M_wdot I_y-M_qdot 0 0;0 0 1 0;0 0 0 1];
A_0 = [Z_w m*U+Z_q 0 0;M_w M_q -z_g*W 0;0 1 0 0;1 0 -U 0];
B_0 = [Z_d;M_d;0;0];

A = inv(M)*A_0;
B = inv(M)*B_0;
C = [0 0 0 1];
D = inv(M)*[-7;0;0;0];
Q1=diag([1,1,4,1]);R1=400;

% Pole Placement -------------------------------------------------------
-
p = [0 -0.6 -0.62 -0.63];
k = place(A,B,p);
klqr = lqr(A,B,Q1,R1)
Ac = A-B*k;
eig(A-B*klqr)
[V,v] = eig(Ac');
s = V(:,4);

% Controller ---smc----------------------------------------------------
------
x0 = [0;0;0;17;0;U;s;k';[0;0;0;3]];                       % initial
condition and command
% Controller lqr
x0 = [0;0;0;17;0;U;0;0;0;0;0;klqr';[0;0;0;3]];
[t,x] = ode45(@remusderivalt_halfpotential,[to tf],x0);

TRUE = 0;
DDIST = 0;
HEIGHT = 0;
R=zeros(length(t),100);
B=R;th=zeros(1,length(t));
for i = 1:length(t)

[xdot,ds,sig,sigdot,h,TRUE,depthcom,range,bearing,threat]=remusderivalt
_halfpotential(t(i),x(i,:)');
    T(i) = TRUE;
    DEP(i) = -depthcom;
    range;
    bearing;
    th(i) = threat;
        for j=1:length(range)
        R(i,j) = range(j);
        B(i,j) = bearing(j);
        end;
    sigma(i) = sig;
    alt(i) = h;
    deltasp(i) = ds*180/pi;
    H(i) = alt(i) + x(i,4);
end;

% Plotting ------------------------------------------------------------
---------
%convert to xy coordinates from rb space
ys=R(:,:).*sin(B(:,:));xs=R.*cos(B);
%plot
  figure(1),clf;
     subplot(2,1,1),plot(x(:,5),DEP,'k--',x(:,5),-x(:,4),'m',x(:,5),-
H,'b'),grid

subplot(2,1,2),plot(x(:,5),deltasp,'m',x(:,5),x(:,3).*180/pi,x(:,5),T*2
0,'k'),grid



     subplot(2,1,1),xlabel('X (m)')
     subplot(2,1,1),ylabel(' Z,H, h (m)')
     subplot(2,1,1),legend('Depth Com','Depth','Seafloor Depth')
     subplot(2,1,2),xlabel('X (m)')
     subplot(2,1,2),ylabel('Rudder Def and Pitch(deg)')
     subplot(2,1,2),legend('Rudder Def','Pitch','True')
     subplot(2,1,1),axis([0 140 -20 0]),HT=gca;set(HT,'FontSize',14)

%figure(2),clf,plot(ys(100,:),xs(100,:),'b*'),grid,hold,axis('equal');

     for i=(1:length(B)-140);
          if (mod(i,10)==0)
               figure(2),clf,plot(xs(i,:),-ys(i,:),'g.'),axis([0,75,-
5,5]),grid,title('Forward Look Blazed Arary Sonar Seafloor
Images'),xlabel('x_s (meters)'),ylabel('y_s (meters)');
           pause;
          end
      end



figure(3),clf,plot(x(:,5),th,'r.'),grid;
title('Threat Level as a Function of Distance Along Track');
```

seabottom.m

```
function H=seabottom(X,S,l);
if (X<= 60), H=20;end;

    if (X>= 60+l), H=20-S;end;
        if (60<X<60+l), H=20-S*.5*(1+tanh((X-60-.5*l)/l));
        end;
```

bweight.m

```
function wb = bweight(b)

b=abs(b)*180/pi;

    if b < 10, wb = 1;end;
    if b > 20,wb = 0;end;
    if b >= 10 & b < 20, wb = 1-(b-10)/10;end;
```

rangeweight.m

```
function w=rangeweight(r)

    if r<0, w=0;end;
    if r == 0, w = 0;end
    if r < 5 & r>0, w = 1;end;
    if r >20,w = 0;end;
    if r >= 5 & r < 20, w = 1-(r-5)/15;end;
```

APPENDIX III: MATLAB CODES FOR HILL/SEA WALL USING FULL POTENTIAL FUNCTION

remusderivalt_hill.m

```
function[xdot,ds,sig,sigdot,h,TRUE,depthcom,range,bearing,threat] =
remusderivalt(t,xx);
%
%
% remusderivalt is an smc controller that is called up by an
% ode function commanding the vehicle to a specific altitude.
% Created by Chris Chuhran, May 1, 2003

% REMUS parameters ----------------------------------------------------

U = xx(6);
s = xx(7:10);
k = xx(11:14)';
x = xx(1:4);         % x(1) = q, x(2) = w, x(3) = theta, x(4) = Z, xx(5)
= X
xcom = xx(15:18);    % xcom = [q com, w com, theta com, depth com]
z_g = 1.96e-2;
x_b = 0;
W = 299;
buoy = 306;

global TRUE;
global DDIST;
global HEIGHT;

I_z = 3.45;
I_y = 3.45;
I_x = 1.77e-1;
U = 1.5;
m = 299/9.81;
M_q = -6.87;
M_qdot = -4.88;
M_w = 30.7;
M_wdot = -1.93;
M_d = -34.6;
Z_q = -9.67;
Z_qdot = -1.93;
Z_w = -66.6;
Z_wdot = -35.5;
Z_d = -50.6;


thetacom = 0;
altcom = 3;

Ra = 20;             % Sonar Range (m)
```

```
%SSTART = 60 - Ra;   % this variable needs to be named once for each
obstacle, hardwired for now
% Dynamics -------------------------------------------------------------
-
M = [m-Z_wdot -Z_qdot 0 0;-M_wdot I_y-M_qdot 0 0;0 0 1 0;0 0 0 1];
A_0 = [Z_w m*U+Z_q 0 0 ;M_w M_q -z_g*W 0;0 1 0 0;1 0 -U 0];
B_0 = [Z_d;M_d;0;0];

A = inv(M)*A_0;
B = inv(M)*B_0;
C = [0 0 0 1];
D = inv(M)*[-7;0;0;0];

% Seafloor Modeling for Sonar (non-time dependent) ---------------------
---
% Seabottom I ----------------------------------------------------------
---
% X_1 = [0:0.5:60];
% X_2 = [60:0.2:62.8];
% X_3 = [62.8:0.5:65.8];
% X_4 = [65.8:0.2:68.6];
% X_5 = [68.6:0.5:111.6];
% X_Model = [X_1 X_2 X_3 X_4 X_5];
%
% H_1 = 20*ones(1,121);
% H_2 = [20:-0.5:13];
% H_3 = 13*ones(1,7);
% H_4 = [13:0.5:20];
% H_5 = 20*ones(1,87);
% H_Model = [H_1 H_2 H_3 H_4 H_5];

% Seabottom II ---------------------------------------------------------
---



X_1 = [0:0.5:60];
X_2 = [60:0.4:61.4];
X_3 = [61.4:0.5:121.4];
X_Model = [X_1 X_2 X_3];
S=6;la=6;
H_Model=seabottom_hill(X_Model,S,la);

bearing=zeros(1,length(X_Model));
range=bearing;

TRUE = 0;
% Sonar ----------------------------------------------------------------
--
for d = 1:length(X_Model)
    if X_Model(d) > xx(5)
       r = sqrt((X_Model(d) - xx(5))^2 + (H_Model(d)-x(4))^2);
        b = asin((H_Model(d) - x(4))/r) ;      % bearing to object as
read by sonar (pitch corrected)
    %
```

```
%         floor_brng = 3*pi/180;         % this is bearing of LF beam
at 3 m alt
%
%         if (b) == 0         % prevents divide by zero error
(sin(angle))
%             floor_alt = 100;             % this happens when obstacle
is directly in front of REMUS, b=0the 100 is arbitrary
%         else
%             floor_alt = 3/sin(b );     % this is range to ocean floor
minus buffer
%         end                                          % buffer of 0.6
can be handled by altitude control
        bearing(d) = 0;range(d) = 0; % initialization
        if (b > -3*pi/180 & b < 17*pi/180 & r < 100)

            %TRUE = 1;  % this means the bottom grid point lies in the
active zone
            bearing(d) = b; range(d) = r; % this registers the range
and bearing of the bottom data point
            %DDIST = r + xx(5) + 0;    % ensures no dive before
obstacle is passed not used

%             for dd = d:length(X_Model)  old stuff from Chris work
%                 if abs(H_Model(dd) - H_Model(dd-1)) <= 0.001
%                     HEIGHT = 20 - H_Model(dd);
%                     break
%                 end
%             end

            %break

        else TRUE = 0;
        end
    end
end


% - Threat Level Assessment


wr = zeros(1,length(range));
wb = wr;
wcount=0;
bcount=0;
for i=1:length(range),
        wr(i) = rangeweight(range(i));if (abs(wr(i))>0),
wcount=wcount+1;end;
        wb(i) = bweight(bearing(i));if
(abs(wb(i))>0),bcount=bcount+1;end;
end;
threat=0;
if (wcount>0 & bcount>0), threat  =
(wr*wr')/wcount*(wb*wb')/bcount;end;
```

```
%  if (xx(3)< -5*pi/180), threat = 0;end; % put in to help on bouncing





% Controller ----------------------------------------------------------
--
% if ((TRUE == 1)| (xx(5) < DDIST))
%
%     %altcom = 3 + (xx(5) - SSTART)*HEIGHT/(35-12);%ramping altitude
command
sig=18;      %controls width of exponential potential function
altcom=3+6*exp(-(xx(5)-60.0)^2/2/sig^2);     %exponential potential
function
    %elseif ((xx(5)-60.0)) > 0 altcom=3;
    %6.0*threat;
% %     if altcom > HEIGHT + 3
% %        altcom = 3 + HEIGHT;
% %     elseif altcom < 3
% %        altcom = 3;
% %     end
% end;

% Seafloor Modeling for Controller (time dependent)
% Seabottom I ---------------------------------------------------------
---
% if xx(5) <= 60 | xx(5) >= 68.6
% H = 20;
% elseif (xx(5) > 60 & xx(5) <= 62.8)
%     H = 170 - 2.5*xx(5);
% elseif xx(5) > 62.8 & xx(5) <= 65.8
%     H = 13;
% elseif xx(5) >65.8 & xx(5) < 68.6
%     H = -151.5 + 2.5*xx(5);
% end

% Seabottom II --------------------------------------------------------
----
% if xx(5) <= 60
%     H = 20;
% elseif (xx(5) > 60 & xx(5) <=61.4)
%     %elseif (xx(5) > 60 & xx(5) <=62)
%      %H = 140 - 2*xx(5);        % depth = 16m
%     H = 320 - 5*xx(5);        % depth = 13m
% elseif xx(5) > 61.4
%     H = 14;
% end
H=seabottom_hill(xx(5),S,la);
depthcom = H - altcom;            % altitude control must be converted
to depth control for EOM

% % if ((TRUE == 1) | (xx(5) < DDIST))          % prevents jump up at
edge
```

```
% %     depthcom = 20 - altcom;                 % hardwired for now,
need to "look back"
% % end

xcom=[0;0;thetacom;depthcom];
 phi = 0.1;
 sig=s'*(x-xcom);
 Nmax= 2;
 %ada = Nmax*0.4/inv((s'*B));

%delta = -k*x-Nmax*0.4*sign(inv((s'*B)))*tanh((sig/phi));
%..........smc controller

xe=(x-xcom);
delta = -k*xe;    %............LQR Controller


if abs(delta) > 0.157                      % REMUS has nine deg max
rudder deflection
    delta = 0.157*sign(delta);
end

h = H - x(4);                              % depth for plotting
purposes
ds = delta;                                % rudder angle for plotting
purposes
xsdot = A*x+B*ds+D;
sigdot = s'*xsdot;
xsdot(4) = [x(1)*cos(x(3))-U*sin(x(3))];   % Large angle approximation
xxdot = [U*cos(x(3))+x(1)*sin(x(3))];      % Horizontal advance
xdot = [xsdot;xxdot;0;0;0;0;0;0;0;0;0;0;0;0;0];
```

REMUSCHRIS_hill.m

```
clear
clc
z_g = 1.96e-2;
x_b = 0;
W = 299;
buoy = 306;
I_z = 3.45;
I_y = 3.45;
I_x = 1.77e-1;
U = 1.5;
to = 0;
tf = 80;

global TRUE;
global DDIST;
global HEIGHT;

TRUE = 0;
DDIST = 0;
HEIGHT = 0;
m = 299/9.81;
M_q = -6.87;
M_qdot = -4.88;
M_w = 30.7;
M_wdot = -1.93;
M_d = -34.6;
Z_q = -9.67;
Z_qdot = -1.93;
Z_w = -66.6;
Z_wdot = -35.5;
Z_d = -50.6;

% Dynamics ------------------------------------------------------------
M = [m-Z_wdot -Z_qdot 0 0;-M_wdot I_y-M_qdot 0 0;0 0 1 0;0 0 0 1];
A_0 = [Z_w m*U+Z_q 0 0;M_w M_q -z_g*W 0;0 1 0 0;1 0 -U 0];
B_0 = [Z_d;M_d;0;0];

A = inv(M)*A_0;
B = inv(M)*B_0;
C = [0 0 0 1];
D = inv(M)*[-7;0;0;0];
Q1=diag([1,1,4,1]);R1=400;

% Pole Placement ------------------------------------------------------
-
p = [0 -0.6 -0.62 -0.63];
k = place(A,B,p);
klqr = lqr(A,B,Q1,R1)
Ac = A-B*k;
eig(A-B*klqr)
[V,v] = eig(Ac');
s = V(:,4);

% Controller ---smc----------------------------------------------------
------
x0 = [0;0;0;17;0;U;s;k';[0;0;0;3]];                       % initial
condition and command
% Controller lqr
x0 = [0;0;0;17;0;U;0;0;0;0;klqr';[0;0;0;3]];
[t,x] = ode45(@remusderivalt_hill,[to tf],x0);

TRUE = 0;
DDIST = 0;
HEIGHT = 0;
R=zeros(length(t),100);
B=R;th=zeros(1,length(t));
for i = 1:length(t)
```

```
[xdot,ds,sig,sigdot,h,TRUE,depthcom,range,bearing,threat]=remusderivalt
_hill(t(i),x(i,:)');
    T(i) = TRUE;
    DEP(i) = -depthcom;
    range;
    bearing;
    th(i) = threat;
        for j=1:length(range)
        R(i,j) = range(j);
        B(i,j) = bearing(j);
        end;
    sigma(i) = sig;
    alt(i) = h;
    deltasp(i) = ds*180/pi;
    H(i) = alt(i) + x(i,4);
end;

% Plotting -----------------------------------------------------------
---------
%convert to xy coordinates from rb space
ys=R(:,:).*sin(B(:,:));xs=R.*cos(B);
%plot
  figure(1),clf;
     subplot(2,1,1),plot(x(:,5),DEP,'k--',x(:,5),-x(:,4),'m',x(:,5),-
H,'b'),grid

subplot(2,1,2),plot(x(:,5),deltasp,'m',x(:,5),x(:,3).*180/pi,x(:,5),T*2
0,'k'),grid


     subplot(2,1,1),xlabel('X (m)')
     subplot(2,1,1),ylabel(' Z,H, h (m)')
     subplot(2,1,1),legend('Depth Com','Depth','Seafloor Depth')
     subplot(2,1,2),xlabel('X (m)')
     subplot(2,1,2),ylabel('Rudder Def and Pitch(deg)')
     subplot(2,1,2),legend('Rudder Def','Pitch','True')
     subplot(2,1,1),axis([0 140 -20 0]),HT=gca;set(HT,'FontSize',14)

%figure(2),clf,plot(ys(100,:),xs(100,:),'b*'),grid,hold,axis('equal');

     for i=(1:length(B)-140);
          if (mod(i,10)==0)
               figure(2),clf,plot(xs(i,:),-ys(i,:),'g.'),axis([0,75,-
9,5]),title('Forward Look Blazed Arary Sonar Seafloor
Images'),xlabel('x_s (meters)'),ylabel('y_s (meters)');
           pause;
          end
      end


figure(3),clf,plot(x(:,5),th,'r.'),grid;
title('Threat Level as a Function of Distance Along Track');
```

seabottom_hill.m

```
function H=seabottom_hill(X,S,l);
H = 20-S*exp(-(X-60).^2/2/.1);






% if (X<= 60), H=20;end;
%
%     if (X>= 60+l), H=20;end;
%         %if (60<X<60+l), H=20-S*.5*(1+tanh((X-60-.5*l)/l));
%         end;
```

bweight.m

```
function wb = bweight(b)

b=abs(b)*180/pi;

    if b < 10, wb = 1;end;
    if b > 20,wb = 0;end;
    if b >= 10 & b < 20, wb = 1-(b-10)/10;end;
```

rangeweight.m

```
function w=rangeweight(r)

    if r<0, w=0;end;
    if r == 0, w = 0;end
    if r < 5 & r>0, w = 1;end;
    if r >20,w = 0;end;
    if r >= 5 & r < 20, w = 1-(r-5)/15;end;
```

LIST OF REFERENCES

Fodrea, Lynn, “Obstacle Avoidance Control for the REMUS Autonomous Underwater Vehicle”, Naval Postgraduate School, December 2002.

Healey, A. J., Dynamics and Control of Mobile Robotic Vehicles (MA-4823), Class Notes, Naval Postgraduate School, CA, Winter 2001.

Churan, Christopher, “Obstacle Avoidance Control for the REMUS Autonomous Underwater Vehicle”, Naval Postgraduate School, September 2003.

Danko, Dale, “Autonomous Underwater Vehicle Obstacle Detection and Assessment using Forward Looking Sonar”, Naval Postgraduate School, September 2004.

Healey, A.J., Kim, J., “Control and Random Searching With Multiple Robots”, Proceedings IEEE CDC Conference 2000, Sydney Australia, November 2000 paper No INV-2303.

Healey, A. J,. “Obstacle Avoidance and Bottom Following for the REMUS Autonomous Underwater Vehicle”, 2004.

Blidberg, Richar D., “The Development of AUVs: A Brief Summary”, Autonomous Undersea Systems Institute, ICRA, Seoul, Korea, May 2001.

Bodman, Samuel W, U.S. Department of Commerce Deputy Secretary, October 2001

Fletcher, Barbara, “The U.S. Navy’s Master Plan: A Vision For UUV Development”, July/August 2001.

RDML Landay, William E. III, Executive Program Officer Littoral and Mine Warfare, Spring 2003.

Jordan, Ken, “REMUS AUV Plays Key Role In Iraq War”, July/August 2003.

www.hydroidinc.com, founded 2001, referenced last May 2005.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Dudley Knox Library
Naval Postgraduate School
Monterey, CA

2. Distinguished Professor Anthony J. Healey, Code ME/HY
Department of Mechanical Engineering
Naval Postgraduate School
Monterey, CA

3. Defense Technical Information Center
Fort Belvoir, Virginia

4. Doug Horner
Naval Postgraduate School
Monterey, CA

5. Dr. Donald Brutzman, Code UW/Br
Undersea Warfare Group
Naval Postgraduate School
Monterey, CA